



Politechnika Wrocławska

**Wydział Informatyki i Zarządzania**

kierunek studiów: Informatyka

specjalność: Projektowanie Systemów Informatycznych

Praca dyplomowa - magisterska

**TITLE**

TITLE EN

Katatzyna Biernat

słowa kluczowe:

KEYWORDS

krótkie streszczenie:

SHORT ABSTRACT

Promotor:	dr inż. Bernadetta Maleszka	.....	.....
	<i>imię i nazwisko</i>	<i>ocena</i>	<i>podpis</i>

Do celów archiwalnych pracę dyplomową zakwalifikowano do:\*

a) kategorii A (akta wieczyste)

b) kategorii BE 50 (po 50 latach podlegające ekspertyzie)

\* niepotrzebne skreślić

pieczęć wydziałowa

Wrocław 2016

Niniejszy dokument został złożony w systemie L<sup>A</sup>T<sub>E</sub>X.

# Spis treści

<b>Todo list</b>	<b>1</b>
<b>Rozdział 1. Cel pracy</b>	<b>3</b>
<b>Rozdział 2. Wstęp</b>	<b>5</b>
<b>Rozdział 3. Przegląd istniejących rozwiązań</b>	<b>7</b>
3.1. Filtrowanie w oparciu o zawartość . . . . .	7
3.1.1. Metody tworzenia profilu użytkownika . . . . .	8
3.1.2. Zalety podejścia content-based . . . . .	8
3.1.3. Najczęściej spotykane problemy . . . . .	8
3.2. Filtrowanie kolaboratywne . . . . .	8
3.2.1. Najczęściej spotykane problemy . . . . .	10
3.3. Popularne serwisy wykorzystujące algorytmy rekomendacji . . . . .	11
3.3.1. Rekomendacja muzyki . . . . .	11
3.3.2. Rekomendacja filmów . . . . .	11
3.3.3. Platformy typu e-commerce . . . . .	11
<b>Rozdział 4. Model systemu</b>	<b>13</b>
<b>Rozdział 5. Algorytmy</b>	<b>15</b>
5.1. Filtrowanie kolaboratywne . . . . .	15
5.1.1. Matrix Factorization . . . . .	16
5.1.2. Biased Matrix Factorization . . . . .	17
5.1.3. SVD++ . . . . .	19
5.2. Filtrowanie z analizą zawartości . . . . .	19
5.2.1. Konstrukcja sieci neuronowej . . . . .	19
5.2.2. Uczenie sieci neuronowej . . . . .	19
5.3. Algorytmy hybrydowe . . . . .	19
5.4. Analiza złożoności i poprawności . . . . .	19
<b>Rozdział 6. Ocena eksperymentalna</b>	<b>21</b>
6.1. Opis metody badawczej . . . . .	21
6.1.1. Miara oceny . . . . .	21
6.1.2. Zbiory danych . . . . .	22
6.2. Środowisko symulacyjne . . . . .	22
6.3. Metodologia . . . . .	23
6.4. Przeprowadzone eksperymenty . . . . .	23

<b>Rozdział 7. Wnioski</b>	<b>25</b>
<b>Rozdział 8. CHAPTER 1</b>	<b>27</b>
8.1. SECTION . . . . .	27
8.2. Section 2 . . . . .	27
8.2.1. Subsection 1 . . . . .	27
<b>Dodatek A. Appendix 1</b>	<b>29</b>
<b>Bibliografia</b>	<b>31</b>

ABSTRACT PL

**Streszczenie**

ABSTRACT EN

**Abstract**



Todo list





## Rozdział 1

# Cel pracy

Celem pracy jest zaproponowanie i zbudowanie hybrydowego algorytmu rekomendacji. Składowymi docelowego algorytmu są metody kolaboratywnego filtrowania oraz metody filtrowania z analizą treści.



## Rozdział 2

# Wstęp

Wraz z rozwojem Internetu zmienił się sposób dostępu do informacji. Kiedyś to użytkownik musiał walczyć pozyskanie wiedzy; dzisiaj to informacje walczą u uwagę użytkowników. W świecie zalanym wiadomościami koniecznym wydaje się być zastosowanie filtra, który odsieje interesującą i wartościową zawartość od tej niechcianej. Tak też z pomocą przychodzą zautomatyzowane mechanizmy rekomendacji.

Jednakże sama idea rekomendacji nie jest niczym nowym. Co więcej, zjawisko to możemy zaobserwować w naturze – na przykład wśród mrówek, które podążają wyznaczoną (rekomendowaną) ścieżką feromonową w poszukiwaniu pożywienia.

Ludzie od niepamiętnych czasów posiłkowali się opiniami innych aby ułatwić sobie dokonanie wyboru, od najbliższego grona znajomych do ekspertów i autorytetów.

Wraz z rozwojem nauk informatycznych problem rekomendacji stał się problemem interesującym badaczy. Za pierwszy system rekomendacji uznaje się *Tapestry* stworzony w laboratoriach Xerox Palo Alto Research Center w 1992 roku. Motywacją było odfiltrowanie rosnącej liczby niechcianej poczty elektronicznej [10].

Wkrótce później idea ta została rozszerzona przez takich graczy jak Amazon, Google, Pandora, Netflix, Youtube, Yahoo etc. aż do formy, jaką znamy dzisiaj: systemu, który sugeruje użytkownikom produkty, filmy, muzykę, strony internetowe na podstawie ich aktywności w sieci [30].

Wielkie koncerny internetowe stale poprawiają jakość swoich algorytmów rekomendacji. Najlepszym przykładem jest tutaj Netflix, który w październiku 2006 zorganizował ogólnodostępny konkurs na najlepszy algorytm. Zadaniem uczestników było ulepszenie algorytmu Cinematch. Już po siedmiu dniach od ogłoszenia konkursu trzy zespoły zdołały przebić Cinematch o 1.06% [21][23]. 18 września 2009 Netflix ogłosił, że zespół BellKor's Pragmatic Chaos poprawił Cinematch o 10,06% osiągając wynik  $RMSE = 0.8567$ . Tym samym wygrał nagrodę w wysokości \$1,000,000 i zakończył konkurs [22][24].

Systemy rekomendacji ulepszone są nieustannie, o czym świadczy chociażby organizowana rokrocznie konferencja *ACM International Conference on Recommender Systems*. Tematyka ta poruszana jest także na konferencjach *European Conference on Information Retrieval*, *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases* i wielu innych. Mimo dużego stopnia

zaawansowania wciąż istnieje pole manewru do ulepszania algorytmów rekomendacji i co za tym idzie zwiększanie zadowolenia użytkowników, które z kolei prowadzi do osiągania korzyści biznesowych.

## Rozdział 3

# Przegląd istniejących rozwiązań

Tradycyjnie wyróżniamy następujące techniki rekomendacji:

- **filtrowanie w oparciu o zawartość** (eng. content-based), technika koncentrująca się na atrybutach elementów. Użytkownikowi rekomendowane są elementy, które podobne są do tych wybieranych przez niego w przeszłości;
- **filtrowanie kolaboratywne** (eng. collaborative filtering), technika polegająca na odnajdywaniu użytkowników o podobnych gustach i sugerowaniu lubianych przez nich elementów aktualnie aktywnemu użytkownikowi;
- **filtrowanie demograficzne** (eng. demographic), technika koncentrująca się na sugerowaniu aktywnemu użytkownikowi elementów popularnych wśród użytkowników z tej samej okolicy bądź w podobnym przedziale wiekowym;
- **filtrowanie z analizą domeny wiedzy** (eng. knowledge-based), technika dobierająca kolejne elementy na podstawie określonej domeny wiedzy na temat tego, jak dany element spełnia potrzeby i preferencje użytkownika;
- **filtrowanie z analizą społecznościową** (eng. community-based), technika dobierająca rekomendacje dla użytkownika w zależności od preferencji innych użytkowników z jego sieci społecznościowej. W myśl zasady ”powiedz mi kim są twoi przyjaciele a powiem ci kim jesteś”;
- **hybrydowe systemy rekomendacji**, to kombinacja dowolnych powyższych technik.

Każda z tych technik ma swoje wady i zalety w zależności od kontekstu, w którym ma być stosowana[26].

### 3.1. Filtrowanie w oparciu o zawartość

Filtrowanie content-based opiera się na cechach elementów w systemie. Rekomendowane są obiekty, które podobne są do tych pozytywnie ocenionych wcześniej przez użytkownika[11]. W zależności od domeny pod uwagę mogą być brane słowa kluczowe, cechy takie jak rok wydania, reżyser, autor, kompozytor, gatunek itp.

### 3.1.1. Metody tworzenia profilu użytkownika

Profil użytkownika może być tworzony na dwa sposoby. Jeżeli użytkownik jawnie pozostawia informacje można mówić o podejściu aktywnym (explicit feedback). Do takich informacji należą: ocena konkretnych elementów, tzw. łapka w górę lub w dół, komentarz itp.

Jednakże nawet jeżeli użytkownik nie jest skory do zostawiania tego typu śladów, to i tak można wiele na jego temat wywnioskować korzystając z podejścia pasywnego (implicit feedback). System bierze wówczas pod uwagę aktywność użytkownika taką jak: historia zakupów, historia przeglądarki a nawet ruchy myszką. W przypadku serwisu z muzyką czy filmem cenną informacją będzie fakt, czy użytkownik wysłuchał lub obejrzał dany materiał do końca czy też wyłączył go po paru sekundach. [18][15]

### 3.1.2. Zalety podejścia content-based

Do zalet filtrowania w oparciu o zawartość należy niezależność użytkownika. Podczas budowania rekomendacji brany pod uwagę jest tylko jego profil; aktywność innych aktorów w systemie nie wpływa na wynik końcowy. Inną przewagą jest przejrzystość – każda propozycja jest w pełni uzasadniona, gdyż opiera się na działaniach użytkownika w przeszłości (podczas gdy w przypadku filtrowania kolaboratywnego mamy do czynienia z czarną skrzynką). Ponadto, tego typu algorytm ma możliwość zaproponowania elementu, który nie był nigdy wcześniej oceniany przez nikogo. Zapobiega to zjawisku długiego ogona [18].

### 3.1.3. Najczęściej spotykane problemy

Aby rekomendacja była skuteczna użytkownik powinien ocenić jak najwięcej elementów. Problematici są zatem użytkownicy, którzy dopiero co dołączyli do serwisu oraz tacy, którzy nie są aktywni i rzadko zostawiają po sobie ślad [19].

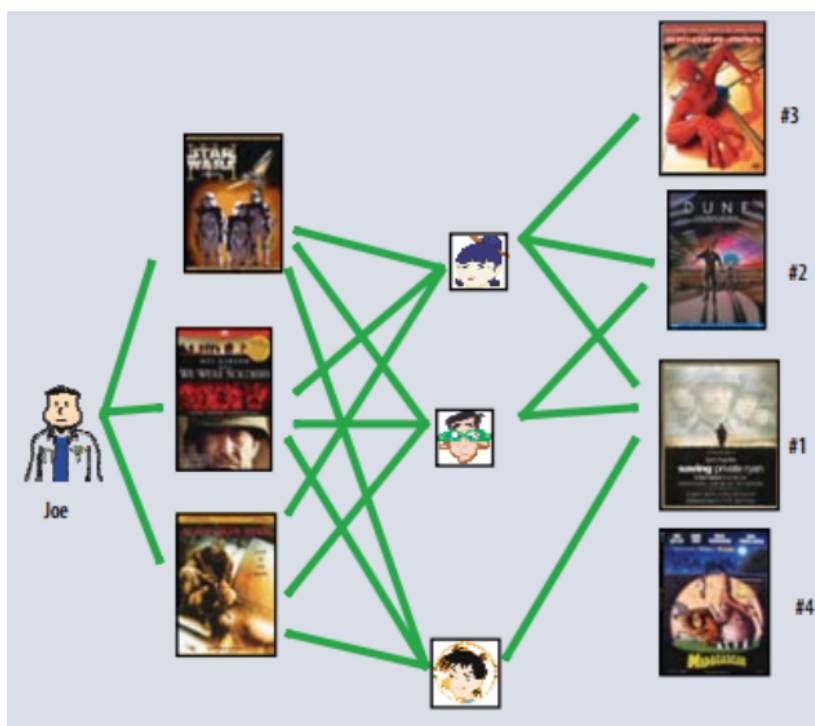
Podejście content-based jest podatne na pułapkę tzw. bańki informacyjnej. Jeżeli w systemie rekomendującym produkcje kinowe użytkownik do tej pory oceniał jedynie filmy akcji, to mało prawdopodobne jest, że algorytm zaproponuje mu ciekawy dramat obyczajowy. Nowe propozycje nie są zaskakujące[18].

## 3.2. Filtrowanie kolaboratywne

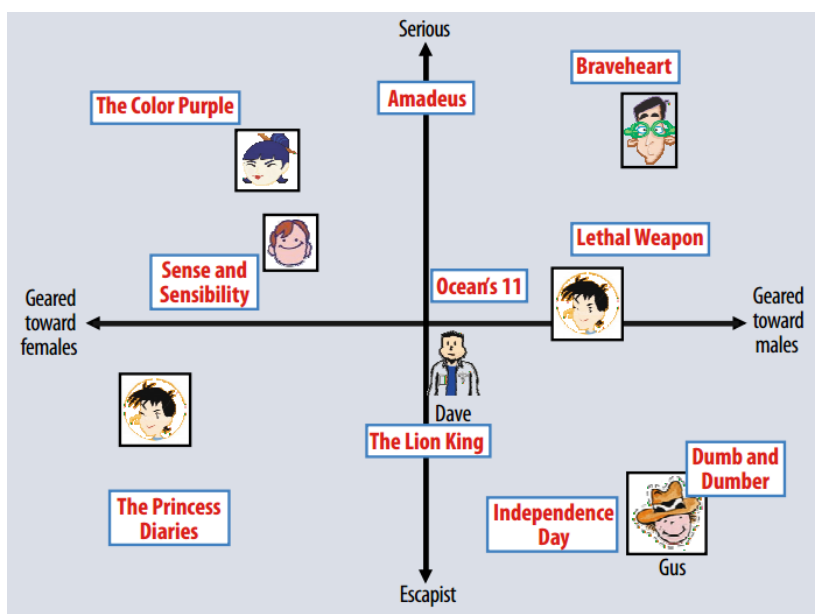
Filtrowanie kolaboratywne opiera się o założenie, że ludzie o zbliżonym guście dokonują podobnych wyborów. Użytkownicy o zbliżonym guście to osoby, które oceniły konkretne elementy w podobny sposób[26][29][11].

W przypadku filtrowania kolaboratywnego można wyróżnić dwa główne podejścia: oparte o regułę sąsiedztwa (ang. *neighborhood*) oraz oparte o model (ang. *model-based*), wykorzystujące modele ukrytych parametrów[16][15].

Filtrowanie w oparciu o regułę sąsiedztwa koncentruje się na związkach element-element bądź użytkownik-użytkownik[15]. Rysunek 3.1 pokazuje regułę sąsiedztwa skoncentrowaną na relacji użytkownik-użytkownik. Joe ocenił trzy filmy. System odnajduje innych użytkowników, którzy ocenili te trzy pozycje podobnie jak Joe. Każdy z nich pozytywnie ocenił film „Saving Private Ryan”, zatem jest to pierwsza rekomendacja dla Joe.



Rys. 3.1: Filtrowanie kolaboratywne metodą sąsiedztwa, zorientowane na użytkownika[16].



Rys. 3.2: Filtrowanie kolaboratywne z wykorzystaniem modelu ukrytych parametrów[16].

Ideą podejścia model-based jest zbadanie i modelowanie zależności element-użytkownik wraz z czynnikami reprezentującymi ukryte własności elementów i użytkowników. Taki

model jest następnie uczony przy użyciu dostępnych danych. W rezultacie można z niego odczytać przewidywaną ocenę elementu dla konkretnego użytkownika[6][15].

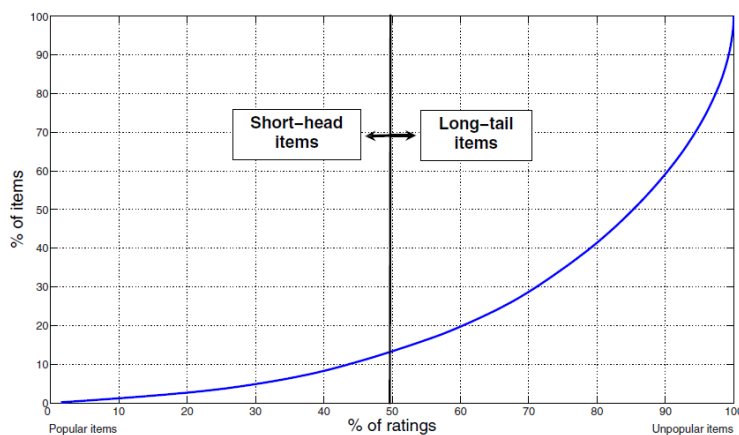
Rysunek 3.2 pokazuje w sposób uproszczony podejście oparte o model. W układzie współrzędnym oznaczeni są użytkownicy wedle swoich preferencji oraz konkretnych cech (np. płeć) a także filmy, które stanowią odpowiedź na dany zestaw preferencji/cech [16].

### 3.2.1. Najczęściej spotykane problemy

Jednym z problemów klasycznego podejścia do kolaboratywnego filtrowania jest brak uwzględnienia dynamiki zmian w gustach użytkowników. Ten sam użytkownik na przestrzeni kilku lat lub miesięcy może zupełnie inaczej ocenić ten sam film bądź piosenkę. Rozwiązaniem jest dodanie czynnika czasu podczas obliczania wag kolejnych ocen. [4][14][16].

Innym problemem jest tzw. zimny start (eng. cold start). Polega on na tym, że użytkownicy nowi w systemie ocenili zbyt mało elementów, aby można było zbudować dla nich dobre rekomendacje[32][28].

Powszechnym zjawiskiem jest tzw. efekt długiego ogona. Rysunek 3.3 przedstawia jak rozkłada się procentowa ilość ocen danych elementów w zależności od ich popularności. Jeżeli algorytm rekomendacji nie wspiera mniej popularnych elementów, to istnieje ryzyko, że użytkownicy nie otrzymają możliwości eksplorowania nowych, niszowych materiałów[28][3].



Rys. 3.3: Problem długiego ogona: 50% ocen dotyczy 10-12% najpopularniejszych elementów w systemie[28].

Systemy rekomendacji wykorzystujące filtrowanie kolaboratywne nie są skalowalne. Złożoność rośnie proporcjonalnie do ilości użytkowników i elementów. Wielkie koncerny internetowe takie jak Twitter wykorzystają klastry i maszyny z bardzo dużą ilością pamięci aby zachować płynność działania serwisu [8].



### 3.3. Popularne serwisy wykorzystujące algorytmy rekomendacji

W przeciągu ostatnich lat algorytmy rekomendacji zagościły na bardzo wielu popularnych serwisach internetowych z różnych domen. Poniższa lista prezentuje garstkę wybranych stron.

#### 3.3.1. Rekomendacja muzyki

- **YouTube** – serwis powstały w 2005 roku, pozwalający na bezpłatne umieszczanie, odtwarzanie, ocenianie i komentowanie filmów. Od 2006 roku przejęty przez Google. YouTube buduje profil użytkownika w oparciu o jego aktywność w serwisie. Brane pod uwagę są polubienia (łapka w górę), subskrypcje, udostępnianie a także informacje czy użytkownik obejrzał film do końca czy tylko pewien jego procent. Techniki rekomendacji stosowane przez serwis to przede wszystkim asocjacyjna eksploracja danych i licznik wspólnych odwiedzin danego wideo w czasie trwania pojedynczej sesji [5].
- **LastFM** – internetowa radiostacja oferująca rozbudowany mechanizm rekomendacji piosenek "Audioscrobber".
- **Pandora** – spersonalizowane radio internetowe wykorzystujące projekt Music Genome Project. Każda piosenka przeanalizowana jest pod kątem maksymalnie 450 cech; na tej podstawie budowane są rekomendacje[1].

#### 3.3.2. Rekomendacja filmów

- **Netflix** – amerykańska platforma oferująca strumieniowanie filmów i seriali. Działający od 2007 roku gigant oferuje rozbudowany system rekomendacji Cinematch[25].
- **Filmweb** – polski serwis poświęcony filmom i jednocześnie druga największa baza filmowa na świecie. Oferuje system rekomendacji Gustomierz, który umożliwia poznawanie nowych filmów w guście użytkownika[7].
- **Internet Movie Database (IMDb)** – największa internetowa baza filmów. Baza zawiera 3,837,014 pozycji, które są oceniane w skali od 1 do 10 przez użytkowników[13].

#### 3.3.3. Platformy typu e-commerce

- **Allegro** – polski portal aukcyjny. Swoim użytkownikom oferuje panel rekomendacji. Prezentowane produkty wybierane są w oparciu o to co dotychczas kupował i oglądał użytkownik[2].
- **Amazon** – największy na świecie sklep internetowy typu B2C. Amazon w swoich mechanizmach rekomendacji wykorzystuje algorytmy filtrowania kolaboratywnego typu item-to-item[17].

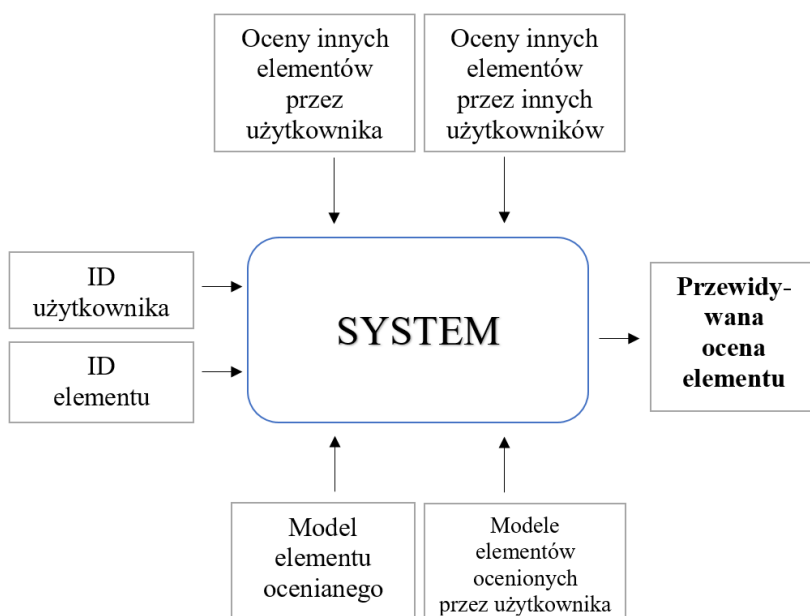


## Rozdział 4

# Model systemu

Głównym założeniem systemu zaproponowanego przez autorkę jest połączenie zalet kolaboratywnego filtrowania i filtrowania w oparciu zawartość minimalizując jednocześnie wady obu podejść.

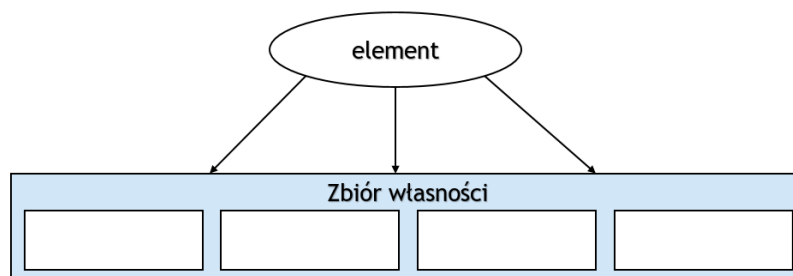
Rys. 4.1 przedstawia czarnoskrzynkowy model systemu. Danymi wejściowymi są numery identyfikacyjne użytkownika dla którego ma być zbudowana rekomendacja oraz elementu, dla którego ma być przewidziana ocena. System pobiera model elementu a także modele wszystkich innych elementów, które użytkownik ocenił w przyszłości. Jednocześnie pobierane są informacje o tym jak użytkownicy systemu ocenili inne elementy. Wynikiem wyjściowym jest predykcja – jak aktywny użytkownik oceni element.



Rys. 4.1: Model czarnoskrzynkowy

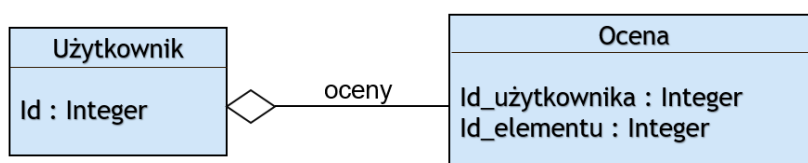
Założeniem systemu jest uniwersalność, zatem model elementu jest uogólniony i dostosowuje się w zależności do domeny, w której system jest wykorzystywany. Rys.

4.2 przedstawia reprezentację elementu w systemie. W zbiorze wartości mogą znaleźć się takie pozycje jak lista aktorów, reżyser (w przypadku filmów), gatunek, wykonawca (w przypadku muzyki), typ produktu lub cena (w przypadku systemów typu e-commerce).



Rys. 4.2: Uogólniony model elementu

Każdy użytkownik systemu jest anonimowy. Nie jest znana jego płeć, wiek, pochodzenie itp. System nie przechowuje także informacji właściwych mediom społecznościowym takich jak relacje między użytkownikami (przyjaźnie, śledzenie). Wiadomym jest jedynie jakie elementy zostały ocenione i jak zostały ocenione. Rys. 4.3 przedstawia reprezentację użytkownika w systemie.



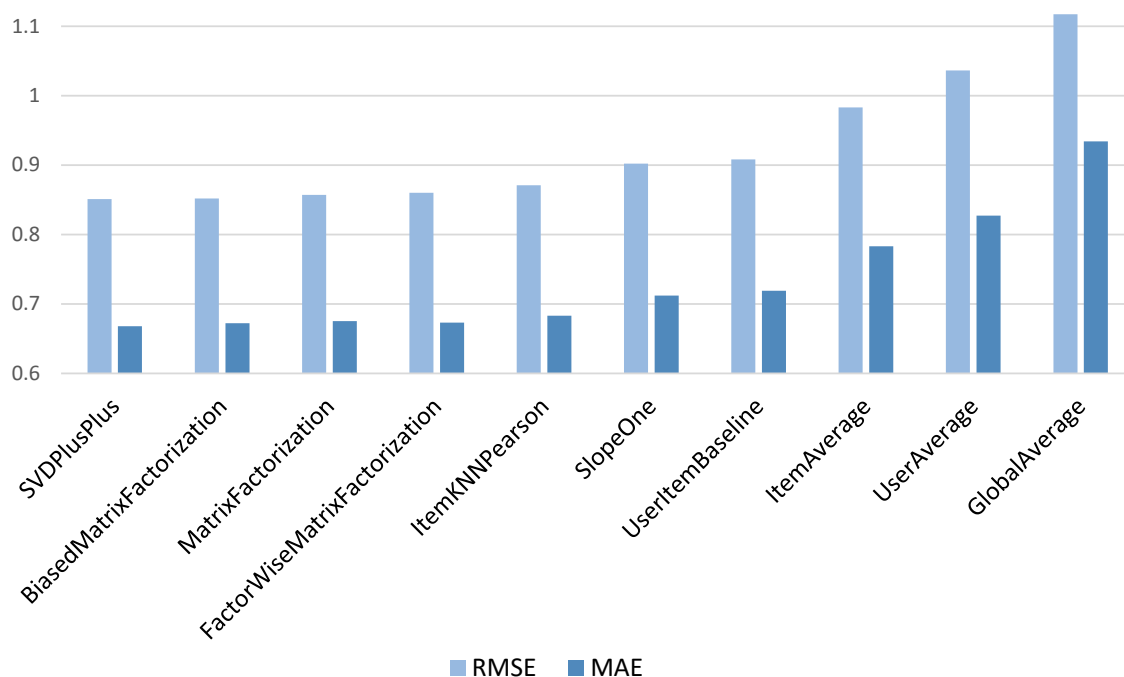
Rys. 4.3: Uogólniony model elementu

## Rozdział 5

# Algorytmy

### 5.1. Filtrowanie kolaboratywne

Przed podjęciem decyzji dotyczącej wyboru algorytmu kolaboratywnego filtrowania zostały przeanalizowane testy na bazie MovieLense M1[9]. Testy przeprowadzone zostały z pięciokrotną walidacją krzyżową. Rys. 5.1 przedstawia wyniki (im mniejsza wartość RMSE i MAE tym lepiej).

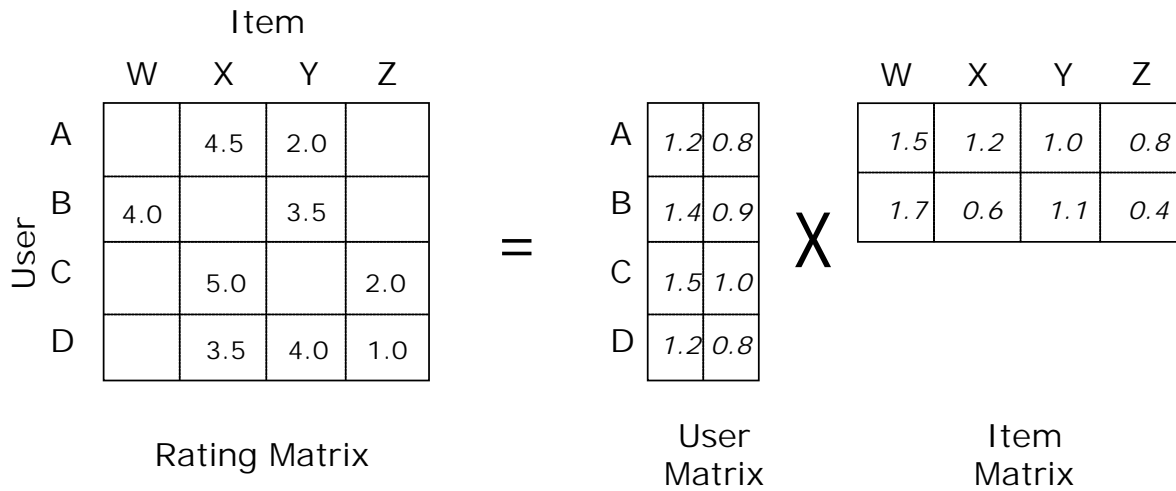


Rys. 5.1: Test algorytmów filtrowania kolaboratywnego [20]

Najefektywniejsze okazały się algorytmy SVD++, Biased Matrix Factorization i Matrix Factorization bazujące na podejściu model-based. One posłużą jako baza do nowego algorytmu rekomendacji.

### 5.1.1. Matrix Factorization

Algorytm *Matrix Factorization* bazuje na matematycznej metodzie rozkładu macierzy na czynniki



Rys. 5.2: Faktoryzacja macierzy [27]

Przebieg algorytmu składa się z dwóch faz. W pierwszej fazie inicjowany jest model (Algorytm 1). Daną wejściową jest macierz zawierająca dotychczasowe oceny wszystkich elementów przez wszystkich użytkowników w systemie. Na wyjściu otrzymywane są dwie nowe macierze reprezentujące ukryte własności użytkowników i elementów. Na tym etapie są one wypełnione wartościami losowymi.

---

#### Algorytm 1 Matrix Factorization – Inicjacja modelu

---

```

 $N \leftarrow$  liczba użytkowników
 $M \leftarrow$  liczba elementów
 $F \leftarrow$  liczba ukrytych własności
 $ratings \leftarrow$  Macierz  $N \times M$  zawierająca dotychczasowe oceny wszystkich elementów
przez wszystkich użytkowników
 $user\_factors \leftarrow$  Macierz  $N \times F$  reprezentująca ukryte własności użytkowników
 $item\_factors \leftarrow$  Macierz  $M \times F$  reprezentująca ukryte własności elementów
for each  $uf \in user\_factors, if \in item\_factors$  do
    wstaw losową wartość za pomocą transformacji Boxa-Mullera
end for
for each  $user, item \in ratings$  do
    if  $ratings_{user,item} = NULL$  then
        wstaw 0 do wiersza  $user\_factors_{user}$  i  $item\_factors_{item}$ 
    end if
end for
return  $user\_factors, item\_factors$ 

```

---

W fazie drugiej następuje uczenie (Algorytm 2). Odbywa się ono metodą stochastycznego gradientu prostego. Wynikiem tej fazy są macierze reprezentujące ukryte

własności użytkowników i elementów. Mnożąc je ze sobą uzyskiwana jest przewidywana ocena każdego z elementów przez użytkowników.

Przed rozpoczęciem uczenia ustalane są parametry: parametr regulujący (*regularization*), tempo uczenia (*learn rate*), parametr zanikania (*decay*) i liczba iteracji. W trakcie trwania głównej pętli parametr regulujący pozostaje niezmienny. Tempo uczenia jest przy każdym przebiegu pętli mnożone przez parametr zanikania, dzięki czemu można kontrolować jak kolejne przebiegi pętli wpływają na finalny wynik.

Ostatnim parametrem ustalonym przed główną pętlą jest skośność globalna (*global bias*), która jest średnią wszystkich znanych ocen.

W pętli uczenia wykonywane są następujące operacje: dla każdej pary użytkownik – element budowana jest przewidywana ocena poprzez obliczenie iloczynu skalarnego odpowiednich wartości z macierzy wartości ukrytych. Ocena ta jest modyfikowana poprzez dodanie globalnej skośności a następnie porównywana z faktyczną oceną elementu przez użytkownika. Tak uzyskany błąd służy do wyliczenia delty. Macierze wartości ukrytych uaktualniane są o wyliczoną deltę.

Pod koniec każdej iteracji aktualizowane jest tempo uczenia.

---

#### Algorytm 2 Matrix Factorization – Faza uczenia

---

```

global_bias ← średnia wszystkich ocen
X ← liczba iteracji
regularization ← parametr regulujący
current_learnrate ← tempo uczenia
decay ← paramert zanikania
for each x ∈ X do
  for each user, item ∈ ratings do
    predicion = global_bias + IloczynSkalarny(user_factorsuser, item_factorsitem);
    error = ratingsuser, item − predicion
    //dopasowanie własności ukrytych:
    for each f ∈ F do
      deltau = error * item_factorsitem, f − regularization * user_factorsuser, f
      deltai = error * user_factorsuser, f − regularization * item_factorsitem, f
      user_factorsuser, f += current_learnrate * deltau
      item_factorsitem, f += current_learnrate * deltai
    end for
  end for
  current_learnrate *= decay
end for
return user_factors, item_factors

```

---

#### 5.1.2. Biased Matrix Factorization

Algorytm *Biased Matrix Factorization* jest modyfikacją wyżej opisanego algorytmu *Matrix Factorization*. Podobnie jak jego pierwowzór składa się z dwóch faz. W pierwszej fazie dodatkowo inicjowane są dwa dodatkowe wektory: skośność użytkowników (*user bias*) i skośność elementów (*item bias*).

Inaczej jest też obliczana skośność globalna:

$$global\_bias = \frac{\frac{a-r_{min}}{r_{max}-r_{min}}}{1 - \frac{a-r_{min}}{r_{max}-r_{min}}}, \quad (5.1)$$

gdzie

- $a$  to średnia wszystkich ocen
- $r_{min}$  to minimalna ocena w systemie
- $r_{max}$  to maksymalna ocena w systemie

Faza druga wygląda podobnie jak w przypadku algorytmu *Matrix Factorization*, jednak są uwzględniane dodatkowe parametry i wykonywane dodatkowe kroki.

---

### Algorytm 3 Biased Matrix Factorization – Faza uczenia

---

```

global_bias ← średnia wszystkich ocen
X ← liczba iteracji
regU, regI, BiasReg ← parametry regulujące dla użytkownika, elementu i ogólny
current_learnrate ← tempo uczenia
BiasLearnRate ← tempo uczenia skośności
decay ← paramert zanikania
for each x ∈ X do
  for each user, item ∈ ratings do
    score = global_bias + user_bias_user + item_bias_item +
    IloczynSkalarny(user_factors_user, item_factors_item)
    sig_score =  $\frac{1}{1+\exp(-score)}$ 
    predicion = rating_min + sig_score + (rating_max - rating_min)
    error = ratings_user,item - predicion
    gradient_common = err * sig_score * (1 - sig_score) * (rating_max - rating_min)
    //dopasowanie skośności:
    user_bias_user += BiasLearnRate * current_learnrate * (gradient_common -
    BiasReg * RegU * user_bias_user)
    item_bias_item += BiasLearnRate * current_learnrate * (gradient_common -
    BiasReg * RegI * item_bias_item)
    //dopasowanie własności ukrytych:
    for each f ∈ F do
      delta_u = gradient_common * item_factors_item,f - RegU * user_factors_user,f
      delta_i = gradient_common * user_factors_user,f - RegI * item_factors_item,f
      user_factors_user,f += current_learnrate * delta_u
      item_factors_item,f += current_learnrate * delta_i
    end for
  end for
  current_learnrate *= decay
end for
return user_factors, item_factors

```

---



### 5.1.3. SVD++

## 5.2. Filtrowanie z analizą zawartości

### 5.2.1. Konstrukcja sieci neuronowej

### 5.2.2. Uczenie sieci neuronowej

## 5.3. Algorytmy hybrydowe

## 5.4. Analiza złożoności i poprawności



## Rozdział 6

# Ocena eksperymentalna

### 6.1. Opis metody badawczej

#### 6.1.1. Miara oceny

W celu zbadania jakości algorytmów zostały zastosowane miary oceny: średnia kwadratowa błędów (RMSE) i średni błąd bezwzględny (MAE).

##### Średnia kwadratowa błędów

Średnia kwadratowa błędów (ang. RMSE – *root mean square error*) jest często wykorzystywaną miarą służącą zmierzeniu różnicy pomiędzy wartościami przewidywanymi a rzeczywistymi (obserwowanymi).

RMSE jest stosunkowo dobrą miarą dokładności ale tylko w celu porównania różnych modeli dla tego samego zestawu danych. RMSE jest zależne od skali, zatem nie sprawdza się najlepiej w przypadku porównywania ze sobą różnych zmiennych [12].

Średnią kwadratową błędów wylicza się ze wzoru:

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (\hat{y}_i - y_i)^2}, \quad (6.1)$$

gdzie

$\hat{y}_i$  to wartość przewidywana

$y_i$  to wartość rzeczywista

Im niższa wartość RMSE tym bardziej zbliżone są wartości przewidywane do rzeczywistych, zatem tym lepszy jakościowo jest model.

##### Średni błąd bezwzględny

Inną miarą mierzenia jakości modeli predykcyjnych jest MAE (ang. *mean absolute error*). Podobnie jak RMSE miara ta jest zależna od skali, zatem najlepiej sprawdza się w działaniu na tym samym zestawie danych [12].

Średni błąd bezwzględny wylicza się ze wzoru:

$$MAE = \frac{1}{n} \sum_{t=1}^n |\hat{y}_t - y_t|, \quad (6.2)$$

gdzie

$\hat{y}_i$  to wartość przewidywana

$y_i$  to wartość rzeczywista

### 6.1.2. Zbiory danych

## 6.2. Środowisko symulacyjne

Rys. 6.1 przedstawia zrzut ekranu środowiska symulacyjnego stworzonego przez autorkę pracy na potrzeby przeprowadzenia badań algorytmów. Główny interfejs programu składa się z trzech części: ustawienia podstawowe, okno z wynikiem oraz panel sterujący ustawieniami zaawansowanymi.

W sekcji z ustawieniami podstawowymi możliwy jest wybór:

- bazy danych, która zostanie wykorzystana do pomiarów;
- typu algorytmu rekomendującego (content-based, collaborative bądź hybrydowy)
- w przypadku wyboru kolaboratywnego filtrowania lub filtrowania hybrydowego możliwy jest wybór typu algorytmu: Matrix Factorization, Biased Matrix Factorization lub SVD++.
- w przypadku wyboru filtrowania z analizą zawartości lub filtrowania hybrydowego automatycznie wybierany jest algorytm oparty na sieci neuronowej.

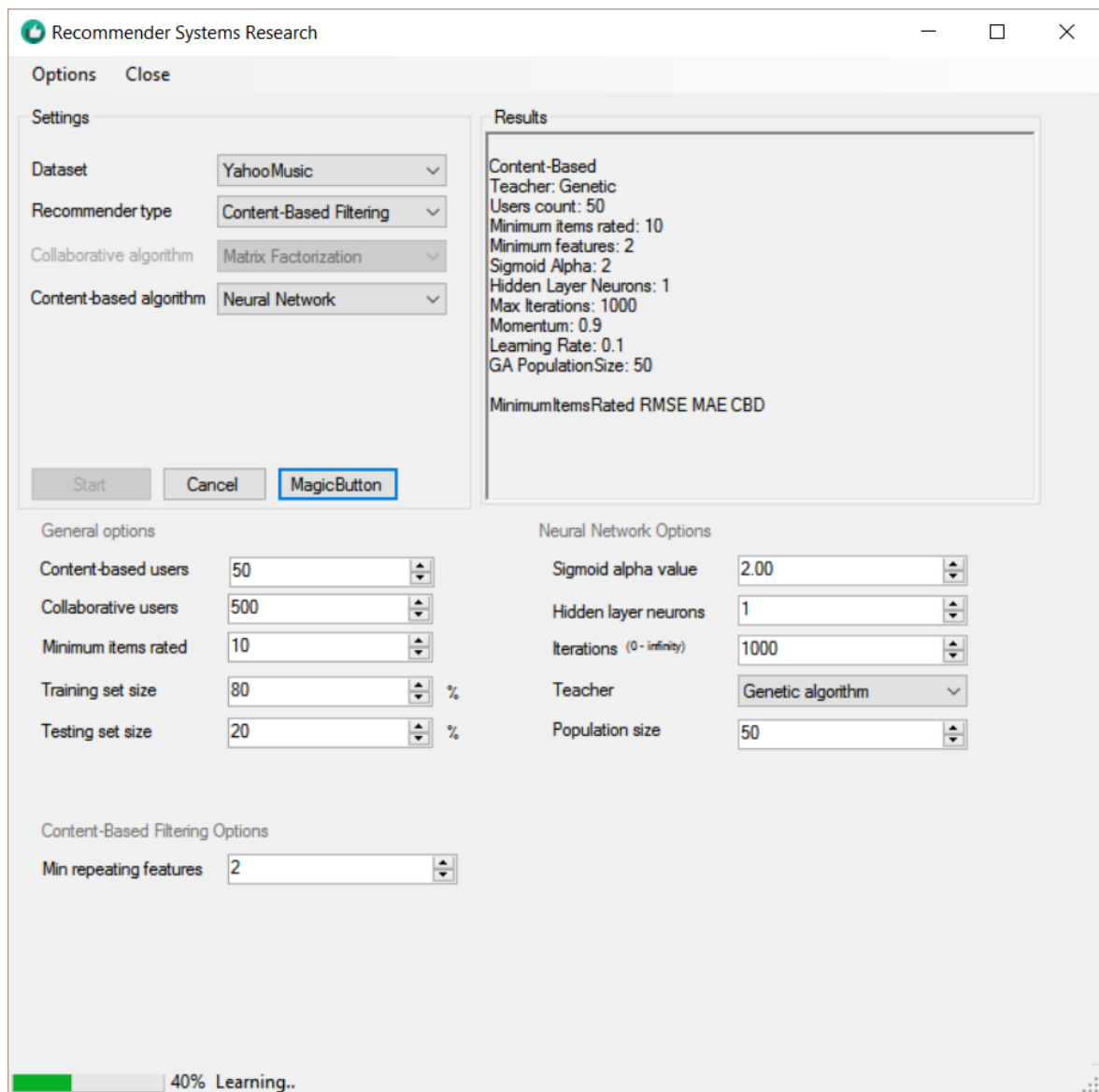
Widok ustawień zaawansowanych zmienia się w zależności od wybranego filtrowania. W przypadku wyboru content-based istnieje możliwość regulowania parametrów sieci neuronowej. Zawsze istnieje możliwość regulowania kryteriów doboru zestawu danych.

Kryteria doboru zestawu danych są następujące:

- liczba użytkowników do pobrania do algorytmów content-based;
- liczba użytkowników do pobrania do algorytmów collaborative;
- minimum elementów, jakie zostały ocenione przez każdego pobranego użytkownika;
- stosunek wielkości zbioru treningowego do zbioru testowego (domyślnie 80%-20%).

Parametry sieci neuronowej podlegające regulacji to:

- Sigmoidalna wartość alfa;
- liczba neuronów w warstwie ukrytej;
- maksymalna liczba iteracji uczenia sieci neuronowej;
- nauczyciel sieci neuronowej: propagacja wsteczna, rprop (resilient backpropagation) lub algorytm genetyczny;
- w przypadku wyboru algorytmu genetycznego – rozmiar każdej kolejnej populacji;
- minimalna ilość powtórzeń danej cechy aby była brana pod uwagę w trakcie budowania rekomendacji.



Rys. 6.1: Zrzut ekranu środowiska symulacyjnego

### 6.3. Metodologia

### 6.4. Przeprowadzone eksperymenty



Rozdział 7

**Wnioski**





Rozdział 8

CHAPTER 1

8.1. SECTION

Algorytm 4

Alghoritm 4

$T \leftarrow$  text under analysis  
**for** each word  $w \in T$  **do**  
     $S_w \leftarrow FIND\_SENTIMENT(w)$   
    **if**  $S_w = POSITIVE$  **then**  
         $Sentiment[POSITIVE]++$   
    **else if**  $S_w = NEGATIVE$  **then**  
         $Sentiment[NEGATIVE]++$   
    **else**  
         $Sentiment[NEUTRAL]++$   
    **end if**  
**end for**  
**return**  $\arg \max_x Sentiment[x]$

Rys. 8.1: Schema 1



8.2. Section 2

8.2.1. Subsection 1

Subsubsection 1  
*Definicja 1*  
*Definicja - pierwsza*



Dodatek A

## Appendix 1

# Spis rysunków

8.1 Schema 1 . . . . .	27
------------------------	----

# Spis wzorów

# Spis algorytmów

1 Matrix Factorization – Inicjacja modelu . . . . .	16
2 Matrix Factorization – Faza uczenia . . . . .	17
3 Biased Matrix Factorization – Faza uczenia . . . . .	18
4 Alghoritm 4 . . . . .	27

# Bibliografia

- [1] About the Music Genome Project. <http://www.pandora.com/about/mgp>. Data dostępu: 2016-07-19.
- [2] Allegro – korzystanie z systemu rekomendacji. <http://faq.allegro.pl/artykul/27613/korzystanie-z-systemu-rekomendacji>. Data dostępu: 2016-07-19.
- [3] Celma O. *The Long Tail in Recommender Systems*, pages 87–107. Springer-Verlag Berlin Heidelberg, 2010.
- [4] Cheng J., Liu Y., Zhang H., Wu X., Chen F. A new recommendation algorithm based on user’s dynamic information in complex social network. *Mathematical Problems in Engineering*, 2015, 2015.
- [5] Davidson J., Liebald B., Liu J., Nandy P., Van Vleet T., Gargi U., Gupta S., He Y., Lambert M., Livingston B. et al. The youtube video recommendation system. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 293–296. ACM, 2010.
- [6] Desrosiers C., Karypis G. *Recommender Systems Handbook*, chapter A Comprehensive Survey of Neighborhood-based Recommendation Methods, pages 107–144. Springer, New York Dordrecht Heidelberg London, 2010.
- [7] Filmweb – najczęściej zadawane pytania. <http://www.filmweb.pl/help>. Data dostępu: 2016-07-19.
- [8] Gupta P., Goel A., Lin J., Sharma A., Wang D., Zadeh R. Wtf: The who to follow service at twitter. In *Proceedings of the 22nd international conference on World Wide Web*, pages 505–514. ACM, 2013.
- [9] Harper F. M., Konstan J. A. The movielens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 5(4):19, 2016.
- [10] Huttner J. From Tapestry to SVD: A survey of the algorithms that power recommender system. Master’s thesis, Haverford College Department of Computer Science, 05 2009.
- [11] Huynh T., Hoang K. Modeling collaborative knowledge of publishing activities for research recommendation. In *International Conference on Computational Collective Intelligence*, pages 41–50. Springer, 2012.

- [12] Hyndman R. J., Koehler A. B. Another look at measures of forecast accuracy. *International journal of forecasting*, 22(4):679–688, 2006.
- [13] IMDb database statistics. <http://www.imdb.com/stats>. Data dostępu: 2016-07-19.
- [14] Ji K., Sun R., Shu W., Li X. Next-song recommendation with temporal dynamics. *Knowledge-Based Systems*, 88:134–143, 2015.
- [15] Koren Y., Bell R. *Recommender Systems Handbook*, chapter Advances in Collaborative Filtering, pages 145–186. Springer, New York Dordrecht Heidelberg London, 2010.
- [16] Koren Y., Bell R., Volinsky C. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.
- [17] Linden G., Smith B., York J. Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing*, 7(1):76–80, 2003.
- [18] Lops P., de Gemmis M., Semeraro G. *Recommender Systems Handbook*, chapter Content-based Recommender Systems: State of the Art and Trends, pages 73–100. Springer, New York Dordrecht Heidelberg London, 2010.
- [19] Maleszka M., Mianowska B., Nguyen N. T. A method for collaborative recommendation using knowledge integration tools and hierarchical structure of user profiles. *Knowledge-Based Systems*, 47:1–13, 2013.
- [20] Mymedialite: Example experiments. <http://www.mymedialite.net/examples/datasets.html>. Data dostępu: 2016-07-24.
- [21] Netflix Prize (I tried to resist, but...). <https://www.snellman.net/blog/archive/2006-10-15-netflix-prize.html>. Data dostępu: 2016-07-08.
- [22] Netflix Prize: forum. <http://www.netflixprize.com/community/viewtopic.php?id=1537>. Data dostępu: 2016-07-08.
- [23] Netflix Prize Rankings. [http://www.hackingnetflix.com/2006/10/netflix\\_prize\\_r.html](http://www.hackingnetflix.com/2006/10/netflix_prize_r.html). Data dostępu: 2016-07-08.
- [24] Netflix Prize Rules. <http://www.netflixprize.com/rules>. Data dostępu: 2016-07-08.
- [25] Pogue D. A Stream of Movies, Sort of Free. *The New York Times*, 2007.
- [26] Ricci F., Rokach L., Shapira B. *Recommender Systems Handbook*, chapter Introduction to Recommender Systems Handbook, pages 1–35. Springer, New York Dordrecht Heidelberg London, 2010.
- [27] Rohrmann T. Computing recommendations at extreme scale with apache flink. <http://data-artisans.com/computing-recommendations-at-extreme-scale-with-apache-flink>, 2015.
- [28] Rubens N., Kaplan D., Sugiyama M. Active learning in recommender systems. In Kantor P., Ricci F., Rokach L., Shapira B., editors, *Recommender Systems Handbook*, pages 735–767. Springer, 2011.
- [29] Schafer J., Frankowski D., Herlocker J., Sen S. *The Adaptive Web*, chapter Collaborative filtering recommender systems, page 291–324. Springer Berlin / Heidelberg, 2007.

- [30] Sharma R., Singh R. Evolution of Recommender Systems from Ancient Times to Modern Era: A Survey. *Indian Journal of Science and Technology*, 9(20), 2016.
- [31] Willmott C. J., Matsuura K. Advantages of the mean absolute error (mae) over the root mean square error (rmse) in assessing average model performance. *Climate research*, 30(1):79–82, 2005.
- [32] Zhang H.-R., Min F., He X., Xu Y.-Y. A hybrid recommender system based on user-recommender interaction. *Mathematical Problems in Engineering*, 2015, 2015.