Cuprins

[1 Introducere 6](#_Toc12218065)

[2 Sisteme de recomandări 7](#_Toc12218066)

[2.1 Content-based recommender 7](#_Toc12218067)

[2.1.1 Avantaje 7](#_Toc12218068)

[2.1.2 Dezavantaje 8](#_Toc12218069)

[2.1.3 Concluzie 8](#_Toc12218070)

[2.2 Collaborative filtering 8](#_Toc12218071)

[2.2.1 Avantaje 9](#_Toc12218072)

[2.2.2 Dezavantaje 9](#_Toc12218073)

[2.2.3 Concluzie 9](#_Toc12218074)

[2.3 Concluzie generală 9](#_Toc12218075)

[3 Implementarea sistemelor de recomandări 10](#_Toc12218076)

[3.1 User-Based Collaborative filtering ( User-User) 10](#_Toc12218077)

[3.1.1 Analiza similarităților 10](#_Toc12218078)

[3.1.2 Rolul KNN (K Nearest Neighbour) 14](#_Toc12218079)

[3.1.3 Predicția unui rating 14](#_Toc12218080)

[3.1.4 Implementare 15](#_Toc12218081)

[3.1.5 Testarea sistemului 16](#_Toc12218082)

[3.2 Item-Based Collaborative Filtering(Item-Item) 19](#_Toc12218083)

[3.2.1 Analiza similarităților 19](#_Toc12218084)

[3.2.2 Predicția ratingurilor 20](#_Toc12218085)

[3.2.3 Implementare 21](#_Toc12218086)

[3.2.4 Testare 21](#_Toc12218087)

[3.2.5 Compararea rezultatelor 23](#_Toc12218088)

[3.3 Content-based recommendations 23](#_Toc12218089)

[3.3.1 Profilul utilizatorului și profilul itemelor 24](#_Toc12218090)

[3.3.2 Recomandările 25](#_Toc12218091)

[3.4 Abordarea hibridă 25](#_Toc12218092)

[4 Arhitectura aplicației și tehnologii utilizate 28](#_Toc12218093)

[4.1 Arhitectura Aplicației 28](#_Toc12218094)

[4.2 Tehnologii utilizate 30](#_Toc12218095)

[4.2.1 Spring Boot 30](#_Toc12218096)

[4.2.2 Spring Data JPA 30](#_Toc12218097)

[4.2.3 Spring security 31](#_Toc12218098)

[4.2.4 Angular 31](#_Toc12218099)

[4.2.5 PostgreSQL 31](#_Toc12218100)

[5 Concluzii finale 33](#_Toc12218101)

[Bibliografie 34](#_Toc12218102)

UNIVERSITATEA ”ALEXANDRU IOAN CUZA” DIN IAȘI

**FACULTATEA DE INFORMATICĂ**

****

LUCRARE DE LICENȚĂ

**Web application for movie recommendations**

**propusă de**

***Pintilei Șerban-Mihai***

**Sesiunea:** iulie,2019

**Coordonator științific**

**Lect.Dr Cristian Frăsinaru**

UNIVERSITATEA ”ALEXANDRU IOAN CUZA” DIN IAȘI

**FACULTATEA DE INFORMATICĂ**

LUCRARE DE LICENȚĂ

**Web application for movie recommendations**

**propusă de**

***Pintilei Șerban-Mihai***

**Sesiunea:** iulie,2019

**Coordonator științific**

**Lect.Dr Cristian Frăsinaru**

**Declarație de consimțământ**

Prin prezenta declar că sunt de acord ca Lucrarea de licență cu titlul „Web Application for Movie Recommendations”, codul sursă al programelor și celelalte conținuturi (grafice, multimedia, date de test etc.) care însoțesc această lucrare să fie utilizate în cadrul Facultății de Informatică.

De asemenea, sunt de acord ca Facultatea de Informatică de la Universitatea „Alexandru Ioan Cuza” din Iași, să utilizeze, modifice, reproducă și să distribuie în scopuri necomerciale programele-calculator, format executabil și sursă, realizate de mine în cadrul prezentei lucrări de licență.

Iași,data

Absolvent Pintilei Șerban-Mihai \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(semnătura în original)

# Introducere

Aflat într-o continuă expansiune,mediul online reprezintă o scurtătură extrem de viabilă pentru cca. 4 miliarde de utilizatori oferind oportunități pentru accesarea de informație, socializare etc. Interacțiunea utilizatorilor si accesarea de informații este realizată prin intermediul aplicațiilor Web.

Motivația alegerii dezvoltării unei aplicații web pentru recomandarea de filme online vine în primul rând din pasiunea pentru vizionarea acestora,adăugând necesitatea unui ”instrument” care să vină în ajutorul utilizatorului care se intreabă ”ce film să vizionez?”. Automat aplicația va trebui să încerce să înglobeze răspunsul la această întrebare, nu în sensul de a decide in locul utilizatorului,cât de a-l ajuta prin reducerea numărului de opțiuni dintr-o gamă vastă de filme, rămânand cele mai potrivite pentru el.Acestea vor fi rezultatul sistemului de recomandări.

Tema este una de tot mai des întalnită,luând în calcul faptul că se pune foarte multe accent pe experiența trăită de utilizator în interacțiunea cu aplicațiile web. Companii precum Amazon,Linkedin,Netflix dezvoltă sisteme de recomandări în încercarea de a ajuta utilizatorii să descopere produse/obiecte/articole cât mai relevante pentru asteptările sau nevoile lor.

Pentru realizarea aplicației vor fi utilizate tehnologii de actualitate, atât pe partea de back-end, cât și pe partea de front-end.

Scopul lucrării constă in dezvoltarea unui sistem de recomandări cât mai eficient,încercând în același timp să ofere un mediu plăcut utilizatorului.

# Sisteme de recomandări

Pe măsura ce dimensiunea si complexitatea aplicațiilor web crește semnificativ, utilizatorul poate pierde timp căutând informațiile de care are nevoie. În același timp, utilizatorul poate să nu știe exact ce caută, sau iși poate schimba preferințele in timp. Sistemele de recomandări furnizează informații personalizate în funcție de profilul utilizatorului, învățând din interacțiunea sa cu aplicația.

## Content-based recommender

Acest sistem recomandă comparând conținutul itemelor[[1]](#footnote-2) cu profilul utilizatorului.Nu sunt implicați alți utilizatori,ci doar bazat pe preferințele proprii , algoritmul va alege articole cu conținut similar.Profilul utilizatorului este reprezentarea structurală a intereselor acestuia,folosit în recomandarea noilor articole.Procesul de recomandare consistă prin urmare în potrivirea caracteristicilor profilului cu conținutul unui articol necunoscut.Astfel, eficiența algoritmului va fi direct proporțională cu acuratețea cu care profilul reprezintă preferințele utilizatorului.

### Avantaje

* Independența față de alți utilizatori – Sistemul de recomandări bazat pe filtrarea de conținut se folosește doar de ratingurile utilizatorului în cauză pentru a modela profilul acestuia,spre deosebire de Collaborative filtering,care va fi prezentat în cele ce urmează.
* Transparență – Modul în care sistemul funcționează poate fi ușor exemplificat prin expunerea atributelor care au condus la adăgarea unui articol în lista de recomandări.Aceste atribute pot fi indicatori pentru credibilitatea sistemului.Prin comparație,sistemele colaborative nu oferă același nivel de transparență,explicația pentru un articol recomandat fiind doar faptul că alți utilizatori necunoscuți cu gusturi asemănătoare au fost interesați de același item.
* În contextul adăugării unui articol nou – Sistemul este capabil să recomande articole care n-au fost încă apreciate de alți utilizatori.În consecință nu apare problema ”primului rating”,care afectează sistemele colaborative care se bazează doar pe preferințele utilizatorului și care nu ar putea recomanda un articol nou până când nu ar strânge un anumit număr de ratinguri din partea celorlalți utilizatori.

### Dezavantaje

* Supra-specializarea – Acest sistem de recomandări nu oferă nici o metodă pentru găsirea neprevăzutului.Numită si problema serendipității,pune în evidență tendința sistemului de a produce recomandări cu un grad scăzut de noutate.
* Problema utilizatorului nou (Cold-Start problem) – Pentru ca sistemul să funcționeze trebuie ca mai întăi să fie creat un profil al utilizatorului,prin colectarea de ratinguri. Prin urmare , când nu sunt suficiente ratinguri disponibile,sistemul nu va putea să ofere recomandări relevante.
* Analiza limitată de conținut – Sistemul nu oferă rezultate conform așteptărilor dacă nu reușește să distingă între articolele care-i plac utilizatorului de cele care nu-i plac.Unele profile ar putea să nu înglobeze anumiți factori care influențează alegerile utilizatorului.

### Concluzie

Sistemul bazat pe filtrarea de conținut este independent de interacțiunea dintre utilizatori dar totuși necesită implicare din parte utilizatorului pentru formarea unui profil cât mai exact care să conducă la recomandări eficiente.

## Collaborative filtering

Filtrarea colaborativă este procesul care se bazează strict pe părerea comunității.Nu se ține cont de însușirile/conținutul unui anumit articol ci doar de ratingurile atribuite acestuia de către comunitate.Se formează o matrice utilizator-articol a notelor,în care fiecare linie reprezintă notele acordate de utilizator ,iar fiecare coloană reprezintă un anumit articol.La interseția dintre o linie si o coloană se va regăsi nota acordată de utilizator.

Există 2 subtipuri de Collaborative filtering : User-Based Collaborative filtering,unde în contextul recomandării unui articol sistemul va căuta utilizatori cu preferințe similare și Item-Based Collaborative filtering,unde sistemul va căuta articole cu note similare în comunitate.

### Avantaje

* Independența de conținut – Sistemul bazat pe filtrarea colaborativă ignoră proprietățile/conținutul articolelor și se focusează doar pe părerea comunității.

### Dezavantaje

* Cold-Start problem – Are nevoie de suficienți utilizatori care să acorde note pentru a putea găsi similarități.
* Problema adăugării unui articol nou – Spre deosebire de filtrarea pe bază de conținut,acest sistem nu poate recomanda un anumit articol dacă acesta nu are un număr minim de note acordate.
* User/Ratings matrix sparsity problem – Deoarce matricea user/rating este una rară ar putea deveni dificil să fie găsiți utilizatori care au acordat note acelorași articole.
* Problema popularității – Sistemul nu poate recomanda articole unui utilizator cu gusturi unice și în același timp tinde să recomande articole populare.

### Concluzie

Sistemul bazat pe filtrarea colaborativă are nevoie de o comunitate cât mai activă din perspectiva acordării de feed-back pentru articole pentru a oferi recomandări cât mai eficiente.

## Concluzie generală

Așadar ambele sisteme de recomandări au plusurile si minusurile lor în ceea ce privește eficiența rezultatelor iar o tehnică de abordare în integrarea unui asemenea sistem într o aplicație web este reprezentată de implementarea unui sistem hibrid.

# Implementarea sistemelor de recomandări

## User-Based Collaborative filtering ( User-User)

Termenul de colaborativ pune în evidență din start fundamentul pe care se bazează acest tip de sistem: ”interacțiunea” colectivă. În contextul livrării unei recomandări către un utilizator vor fi căutați utilizatori cu gusturi și preferințe similare.

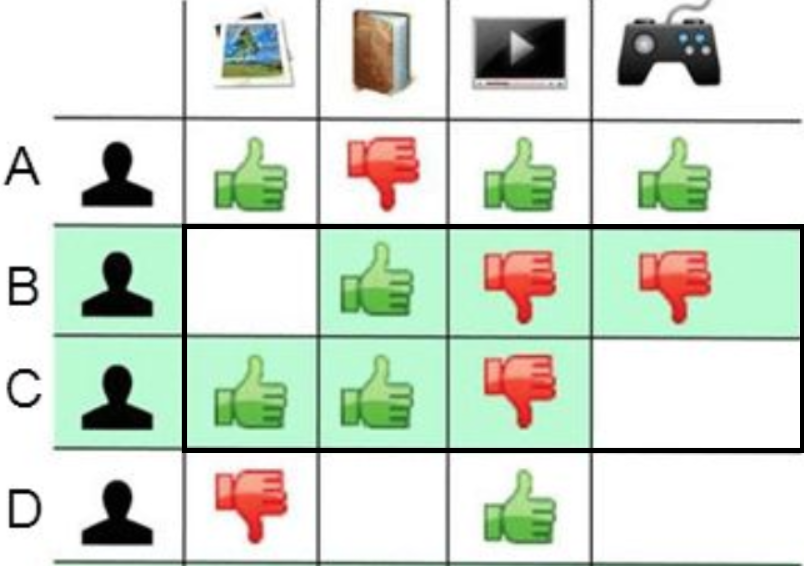


Figura 1 : similaritatea dintre utilizatori în matricea user-item

### Analiza similarităților

Analizând din perspectiva matricii user-item putem defini ca similaritatea dintre doi utilizatori interpretarea ratingurilor comune date de aceștia.Spre exemplu,în Figura 1 nivelul de similaritate dintre utilizatorii B și C este unul maxim , luând în calcul faptul că articolele pe care amândoi le-au evaluat au primit aceeași valoare.

Pentru estimarea similarității dintre utilizatori există diferite tipuri de măsurători care vor fi folosite în experimentul de testare al sistemului cu scopul de a găsi cea care se potrivește cel mai bine pentru sistemul dezvoltat.

Imaginând un spațiu 2D care reprezintă notele acordate de către utilizatori pentru două filme,pe fiecare dintre axe fiind prezente ratingurile posibile (de la 1 la 5) .

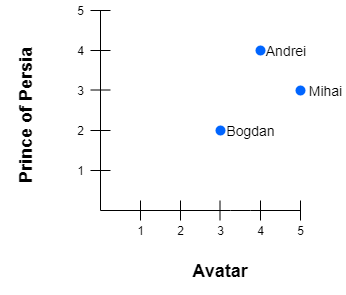
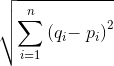


Figura 2 : spațiu 2D pentru reprezentarea ratingurilor

Distanța dintre Bogdan și Mihai va reprezenta similaritatea la nivel de preferințe dintre cei doi. Desigur, această distanță trebuie calculată într-un spațiu cu N filme , deci N-dimensional.

(1)



Aceasta poate fi calculată cu ajutorul distanței Euclidiene, unde qi si pi reprezintă ratingurile filmului comun i.

O problemă care poate fi luată în calcul este non-concordanța dintre modurile de acordare a notelor de către utilizatori.Pot fi utilizatori pentru care nota 4 înseamnă un film foarte bun și utilizatori care notează articolele care le-au plăcut cu nota maximă iar celelalte cu 4.Pot fi utilizatori care notează doar cu nota minimă si nota maximă.

Una dintre soluțiile la aceasta problemă este utilizarea coeficientului de corelație Pearson.Acesta descrie relația dintre două variabile si are o valoarea reală cuprinsă în intervalul [-1,+1] , unde -1 semnifică că una dintre variabile are tendința sa descrească când cealaltă crește, 0 faptul că nu există nici un fel de relație între cele două variabile și 1 dacă o variabilă are tendința să crească o dată cu cealaltă.

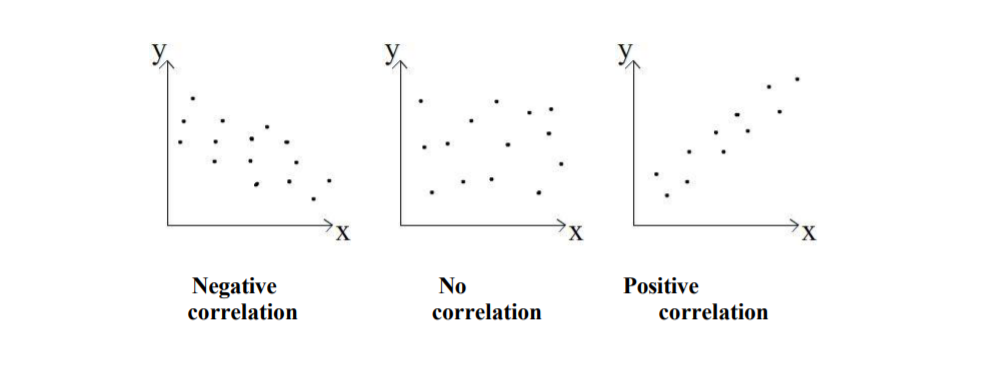
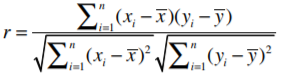


Figura 3 : corelația Pearson

(2)



(2) Reprezintă formula de obținere a coeficientului de corelatie Pearson,xi și yi reprezintă ratingurile acordate filmului comun i,iar și mediile ratingurilor comune acordate de utilizatorii x și y. Prin scăderea valorii medii se obține centrarea datelor,deoarce ratingurile sunt valori pozitive ,prin acest procedeu coeficientul de similaritate va avea valori cuprinse în intervalul [-1,1].

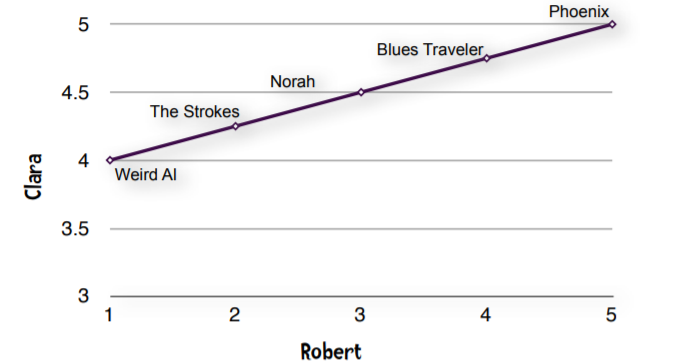
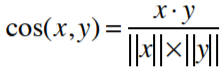


Figura 4 : corelație perfectă

În Figura 4 se poate observa o corelație perfectă între notele acordate de utilizatorii Clara și Robert.

 (3)

Prin similaritatea cosinusului (3) se calculeaza cosinusul unghiului dintre doi vectori nenuli.Valoarea numărătorului semnifică produsul scalar dintre cei doi vectori ,iar la numitor se regăsește produsul dintre lungimile(modulele) celor doi vectori date de formula(4).

modul-vector.png(4)

Se poate observa asemănarea dintre formulele de calcul ale coeficientului de similaritate Pearson (2) și similaritatea cosinusului (3),cu deosebirea că cea din urmă nu centrează datele.

likelihood1.gif(5)

loglikelihood.gif(6)

Relația (5) exprimă verosimilitatea(likelihoodul) în exprimarea parametrilor.Prin date se înțelege datele avute la dispoziție iar prin θ setul de parametri care influențează datele.Pentru ușurința calculelor se folosește log-likelihoodul deoarece logaritmul este o funcție monoton crescătoare,acest lucru însemnând că se păstrează relevanța rezultatului.

jaccardindex.png(7)

Formula (7) reprezintă modul de calculare al indexului Jaccard , cunoscut ca și raportul dintre interescția și reuniunea a două mulțimi,acestea reprezentând mulțimile ratingurilor acordate de doi utilizatori pentru care se vrea calculată similaritatea.De observat este faptul că indexul Jaccard nu ține cont de valoarea ratingurilor ci doar de prezența acestora.

### Rolul KNN (K Nearest Neighbour)

K-Nearest-Neighbour reprezintă unul din cei mai populari algoritmi de Machine Learning utilizați pentru clasificare.Acesta are un rol important și în algoritmul de filtrare colaborativă bazat pe similaritățile dintre utilizatori.Acestea sunt calculate pentru fiecare dintre utilizatori,doi câte doi ,iar la final vor fi aflate toate similaritățile dintre ei.Însă în contextul efectuării unei recomandări de către sistem ar trebui luate în calcul preferințele mai multor utilizatori similari cu cel pentru care se recomandă un articol și nu doar a celui cu nivelul de similaritate cel mai bun.Pentru acest lucru va fi utilizat algoritmul KNN , unde K va reprezenta numărul de utilizatori similari al căror preferințe vor fi luate în calcul pentru elaborarea de recomandări pentru utilizatorul curent.

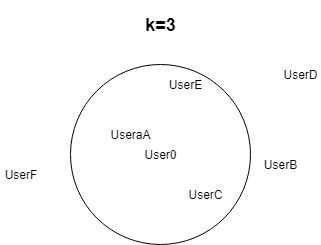
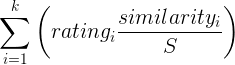


Figura 5 : cei mai similari utilizatori pentru k=3

În Figura 5 sunt exprimate similaritățile utilizatorilor A,B,C,D,E,F cu utilizatorul notat User0.Pentru k=3 vor fi luate în calcul preferințele celor mai apropiați 3 utilizatori în elaborarea recomandărilor pentru User0.

### Predicția unui rating

Luând în considerare similaritățile cu ceilalți utilizatori , precum și care sunt cele mai apropiate K similarități se poate realiza predicția ratingului unui anumit film.

(8)

Ratingul estimat al unui anumit film va fi egal cu rezultatul prezentat în formula (8),unde k reprezintă numărul de utilizatori similari luați în calcul,ratingi reprezintă ratingul acordat de utilizatorul i ,similarityi reprezintă valoare similarității dintre utilizatori,iar S reprezintă suma tuturor celor k similarități ale utilizatorilor implicați.

### Implementare

Pentru implementarea sistemului a fost folosit framework-ul de la Apache Mahout,care oferă suport pentru statistică și machine learning.



Figura 6 : dependințe Mahout

dataModel.png

Figura 7 : DataModel

DataModel stochează preferințele utilizatorilor,și poate primi ca input un fișier în care pe fiecare linie un rating este exprimat ca userID,itemID,rating , sau o bază de date la care trebuie specificate coloanele userID,itemID și rating de unde pot fi preluate datele.

usersimilarity.png

Figura 8 : UserSimilarity

UserSimilarity exprimă metoda de calculare a similarităților dintre utilizatori care primește modelul.

userneihgbourhood.png

Figura 9 : UserNeighbourHood

UserNeighbourHood primește ca input o valoare n echivalentă cu K-ul care stabilește vecinătatea de selecție a celor mai apropiate similarități,tipul de măsurare al similarităților și si modelul.

userrecommender.png

Figura 10 : UserBasedRecommender

UserBasedRecommender primește ca parametrii modelul,vecinătatea precum și modul de calculare al similarităților.

userbased_estimatepref.png

Figura 11 : predicția unui rating

Prin metoda estimatePreference care primește ca parametrii un userID si un itemID se efectuează predicția ratingului pentru itemul respectiv.

userbased_recommend.png

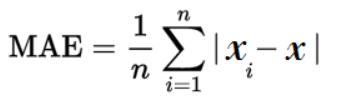
Figura 12 : elaborarea recomandărilor

Prin metoda recommend sistemul va estima preferințele pentru toate itemele cărora utilizatorul cu id-ul dat ca prim parametru nu le-a acordat o notă și va returna o listă de lugime cel mult egală cu al doilea parametru, o listă de obiecte RecommendedItem care conțin id-ul itemului respectiv și ratingul prezis ordonate descrescător în funcție de valoarea ratingului.

### Testarea sistemului

Pentru a putea stabili ce tip de similaritate precum si ce vecinătate trebuie aleasă pentru un sistem de recomandări cât mai precis din punct de vedere al filmelor recomandate trebuie efectuate câteva teste.

Pentru evaluare a fost folosit dataset-ul MovieLens 100K Dataset [[2]](#footnote-3) care prevede 5 perechi de fișiere cu date de antrenament(80% din date) – fișiere cu date de test(20% din date) disjuncte care însumează 100.000 ratinguri acordate de către 1000 utilizatori penru 1700 de filme. Aceste perechi vor fi utilizate pentru 5 fold cross validation[[3]](#footnote-4).

(9)

Metrica de măsurare a acurateții sistemului a fost MAE(Mean absolute error)(9).După colectarea datelor din fișieirul de antrenament au fost comparate ratingurile prezise de sistem cu cele existente în fișierul de test.Media absolută a erorii a fost calculată pentru fiecare utilizator în parte iar formula exprimă de fapt media diferențelor dintre ratingul prezis și cel adevărat.

eroare_sistem.gif(10)

Eroarea totală a sistemului este reprezentată în relația (10) și semnifică raportul dintre suma MAE a tuturor utilizatorilor și numarul de utilizatori.

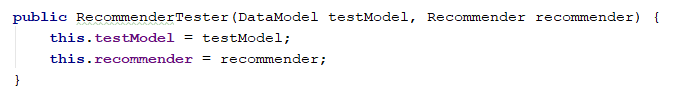


Figura 13 : constructor RecommenderTester

Testarea a fost realizată prin intermediul clasei RecommenderTester(Figura 13) care primește în constructor un set de date de test de tip DataModel și un sistem de recomandări.

Datele prezentate în continuare reprezintă mediile rezultatelor după efectuarea celor 5 teste , realizate pentru fiecare tip de similaritate si dimensiune a vecinătății în parte.Testele au conținut următoarele date :

|  |  |  |
| --- | --- | --- |
|  | Nr.Utilizatori | Nr.iteme |
| Test nr.1 | 459 | 1410 |
| Test nr.2 | 653 | 1420 |
| Test nr.3 | 869 | 1423 |
| Test nr.4 | 923 | 1394 |
| Test nr.5 | 927 | 1407 |

În primul tabel a fost aleasă dimensiunea vecinătății K=5,în al doilea K=20 iar în ultimul K=50.

|  |  |  |  |
| --- | --- | --- | --- |
| Tip de similaritate | Nr. utilizatori cu predicții inevaluabile | MAE |  |
| Pearson correlation | 273 | 483.8733346 | 0.9827395724 |
| Cosine similarity | 363 | 382.0895156 | 0.9464506068 |
| Jaccard index | 28.6 | 626.0816718 | 0.8729258474 |
| LogLikelihood | 13.4 | 658.1567776 | 0.8759641294 |
| Euclidean distance | 336.2 | 405.025843 | 0.9431045046 |

Pentru K=5 se poate observa că similaritatea cosinusului,distanța euclidiană si coeficientul de corelație Pearson au probleme în estimarea predicțiilor pentru foarte mulți utilizatori.

|  |  |  |  |
| --- | --- | --- | --- |
| Tip de similaritate | Nr. utilizatori cu predicții inevaluabile | MAE |  |
| Pearson correlation | 75.6 | 647.1083922 | 0.9213787618 |
| Cosine similarity | 113.6 | 610.3765668 | 0.932070206 |
| Jaccard index | 8 | 642.4743778 | 0.8479146246 |
| LogLikelihood | 1.8 | 648.0701866 | 0.84832108 |
| Euclidean distance | 88 | 592.7175372 | 0.8690639148 |

Pentru o vecinătate mărită K=20 se observă o îmbunătățire atât la nivelul utilizatorilor cu predicții incomplete cât și la eroare sistemului.

|  |  |  |  |
| --- | --- | --- | --- |
| Tip de similaritate | Nr. utilizatori cu predicții inevaluabile | MAE |  |
| Pearson correlation | 14 | 666.9630926 | 0.8877999626 |
| Cosine similarity | 42.8 | 658.2490688 | 0.9061013576 |
| Jaccard index | 2.6 | 635.3860742 | 0.8324840872 |
| LogLikelihood | 0.8 | 637.5444842 | 0.8333614334 |
| Euclidean distance | 17 | 628.1904238 | 0.8380871434 |

Pentru K=50 se observă că eroare sistemului este cea mai mică pentru toate similaritățile,putem deduce astfel că rata erorii este invers proporțională cu K.Acest lucru se datorează faptului că într-o vecinătate mică pot fi înglobați utilizatori care n-au acordat încă un rating pentru filmul pentru care se efectuează predicția.

## Item-Based Collaborative Filtering(Item-Item)

Spre deosebire de filtrarea colaborativă bazată pe compararea similarităților dintre utilizatori,abordarea Item-Item propune compararea similarităților dintre articole.Dacă în cazul celei dintâi similaritățile se calculau comparând conținutul de pe linii din reprezentarea matricii User-Item,acum acestea se vor calcula comparând conținutul coloanelor.

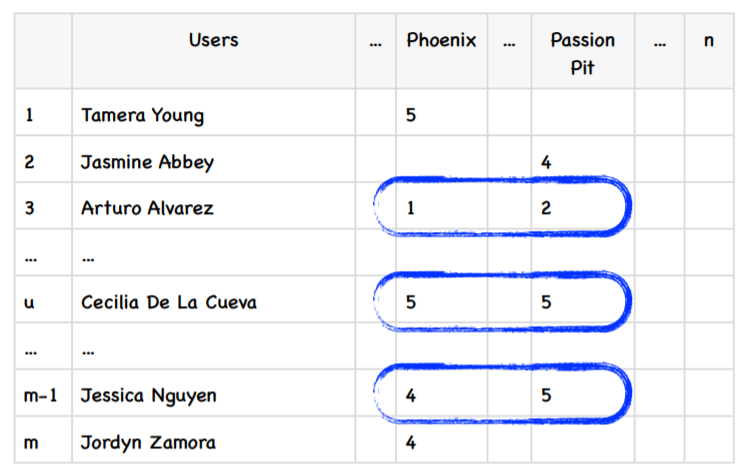
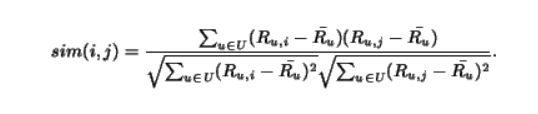


Figura 14 : Item-based Collaborative Filtering

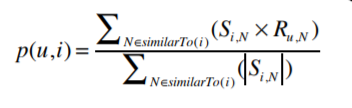
### Analiza similarităților

Distanța euclidiană,indexul Jaccard,log-verosimilitatea și similaritatea cosinusului vor fi păstrate pentru analiza sistemului.Acestea vor fi utilizate însă cu valorile de pe coloane spre deosebire de filtrarea User-User care se folosea de valorile de pe linii. Coeficientul de corelație Pearson este înlocuit însă de similaritatea cosinusului ajustată.

(11)

Necesitatea unei asemenea formule vine din cauza că ar putea fi nevoie în interpretarea similarităților și de diferența dintre modurile în care utilizatorii acordă note.Astfel,din valoarea fiecărui rating va fi scăzută media ratingurilor acordate de utilizatorii care au notat atât articolul i cât și itemul j. Această problemă a fost discutată și la analiza coeficientului de corelație Pearson.

### Predicția ratingurilor

(12)

Modul de prezicere a ratingurilor este exprimat de relatia(12).Pentru utilizatorul u și itemul i pentru care se calculează predicția se însumează produsele dintre toate itemele care sunt similare cu i și ratingul lor ,împărțind rezultatul la suma modulelor tuturor similarităților implicate.Este de menționat că ratingurile vor fi normalizate(13) înainte de a efectua calculul pentru a avea un rezultat în intervalul [-1,1].

normalization.gif(13)

Se poate observa că dacă un anumit item are o similaritate negativă cu itemul i dar a i-a fost acordată o notă slabă aportul adus predicției va fi unul pozitiv,la fel cum o similaritate bună si un rating bun va duce la același rezultat.Acest lucru în contextul prezicerii ratingului unui film i că daca utilizatorului nu i-a plăcut un film care nu este similar cu i atunci ,sau i-a plăcut un film care este similar cu atunci există șansa să-i placă și filmul in cauză.

denormalization.gif(14)

Ratingul prezis pentru itemul i va fi egal cu p denormalizat cu ajutorul formulei (14)(deoarece valoarea lui p va fi în intervalul [-1,1]) iar sistemul trebuie să ofere un rating în intervalul [minrating,maxrating].

### Implementare

Nu vor fi evaluate toate similaritățile dintre articole ,cum se proceda la sistemul colaborativ User-User unde aveam nevoie să formăm vecinătatea bazată pe cele mai bune similarități.Pentru utilizatorul U și itemul i pentru care se dorește efectuarea unei predicții vor fi evaluate doar similariățile dintre itemul pentru care se dorește prezicerea ratingului și celelalte iteme pe care utilizatorul le-a notat.Așadar pe măsură ce vor fi calculate similaritățile dintre filme se va calcula concomitent predicția.Acest lucru optimizează sistemul și oferă un avantaj clar în scalabilite față de sistemul prezentat anterior.În plus,dacă este luată în calcul scalabilitatea sistemelor , metoda de filtrare Item-Item este net superioară metodei User-User deoarece în contextul filmelor într-o aplicație dezvoltată numărul utilizatorilor va fi cu siguranță mai mare decat numărul filmelor.

### Testare

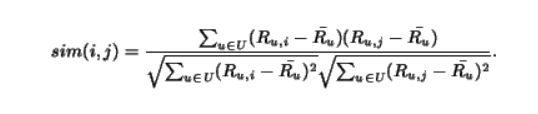
Pentru testarea sistemului va fi utilizată aceeași metrică prezentată anterior,Mean absolute error.Aceasta va fi evaluată de clasa RecommenderTester.Eșantionul inițial de teste este același(Datasetul Movielens 100k[[4]](#footnote-5)).Vor fi evidențiate mediile rezultatelor din cele 5 teste.

|  |  |  |
| --- | --- | --- |
|  | Nr.Utilizatori | Nr.iteme |
| Test nr.1 | 459 | 1410 |
| Test nr.2 | 653 | 1420 |
| Test nr.3 | 869 | 1423 |
| Test nr.4 | 923 | 1394 |
| Test nr.5 | 927 | 1407 |

De menționat este faptul că nu vor mai fi testate diferite valori ale lui K ,pentru că nu mai este utilizat conceptul de vecinătate.

|  |  |  |  |
| --- | --- | --- | --- |
| Tip de similaritate | Nr. utilizatori cu predicții inevaluabile | MAE |  |
| Adjusted Cosine | 0.2 | 648.5446266 | 0.8467653476 |
| Cosine similarity | 0 | 646.8644212 | 0.8443689666 |
| Jaccard index | 0 | 623.2359026 | 0.8134985716 |
| LogLikelihood | 0 | 636.8166964 | 0.8314751792 |
| Euclidean distance | 0 | 637.9433212 | 0.8327956654 |

Se poate observa faptul că nu au mai existat utilizatori cu predicții inevaluabile.În cazul similarității ajustată a cosinusului a existat un test din setul de teste în care a fost nevoie de prezicerea notei pentru un film care avea în setul de antrenament doar un singur utilizator care i-a acordat notă.Acel utilizator și utilizatorul pentru care se cerea predicția aveau în comun un film pentru care au acordat rating.Raportul de similaritate care se realiza între itemele pe care utilizatorul pentru care se cerea predicția si itemul efectiv cerut se reducea mereu la compararea unui singur rating.



Astfel mediile ratingurilor Ru ar avea valorile egale cu valoarea Ru,i și Ru,j și numitorul rezultat va avea valoarea 0.

Deoarece rezultatele au fost pozitive s-a mai efectuat un test pe un set mare de date.A fost utilizat Datasetul MovieLens 10M movie ratings[[5]](#footnote-6) care conține 10 milioane de ratinguri acordate de 72.000 de utilizatori pentru 10.000 de filme.Testarea a fost efectuată pentru a vedea cum se comportă algoritmul pentru un set mare de date.

|  |  |  |  |
| --- | --- | --- | --- |
| Tip de similaritate | Nr. utilizatori cu predicții inevaluabile | MAE |  |
| Adjusted Cosine | 0 | 83544.42567 | 1.1955755126 |
| Cosine similarity | 0 | 83432.49808 | 1.1939737554 |
| Jaccard index | 0 | 82305.91881 | 1.1778516674 |
| LogLikelihood | 0 | 82523.46327 | 1.1809648714 |
| Euclidean distance | 0 | 82643.27778 | 1.1826794955 |

Se poate observa că ierarhia în funcție de eroarea gasită se pastrează,indexul Jaccard fiind în continuare similaritatea care dă randamentul cel mai bun.S-a decis așadar utilizarea acestei similarități în implementarea sistemului final,fiind cea mai potrivită atât în startul aplicației cât și în eventualitatea dezvoltării acesteia pe termen lung.

### Compararea rezultatelor

Comparând rezultatele testelor efectuate pe sistemele de filtrare colaborativă se poate observa că cel din urmă evaluat are rezultate mai bune.

|  |  |  |  |
| --- | --- | --- | --- |
| Sistem colaborativ | Similaritate | Nr.utilizatori cu predicții inevaluabile |  |
| Item-Item | Jaccard index | 0 | 0.8134985716 |
| Item-Item | LogLikelihood | 0 | 0.8314751792 |
| Item-Item | Euclidean distance | 0 | 0.8327956654 |
| User-User | LogLikelihood | 0.8 | 0.8333614334 |
| User-User | Jaccard index | 2.6 | 0.8324840872 |

Este de menționat și faptul că timpul de execuție în rularea testelor a fost de aproximativ 5 ori mai bun în cazul sistemului colaborativ Item-Item . Durata de execuție a testului nr.5 a fost de ~2 minute in cazul ultimului sistem prezentat și ~10 minute în cazul sistemului User-User.

## Content-based recommendations

Sistemele de recomandări bazate pe conținut au ca scop recomandarea de articole cu conținut similar celui care i-a plăcut utilizatorului. Spre deosebire de sistemele bazate filtrarea colaborativă nu este luată în calcul părerea comunității. Ideea de bază constă într-o evaluare cât mai exactă a caracteristicilor care-i plac utilizatorului la un anumit articol,și utilizarea acestora în recomandarea de articole noi.(Figura 16)

Sistemul înregistrează preferințele utilizatorului ca o listă de caracteristici.Bazate pe aceastea va construi profilul utilizatorului pentru a putea să-i cunoască gusturile.Aceste caracteristici denumite în mod clasic feature[[6]](#footnote-7)-uri pot fi privite ca un vector cu valori reale sau boolene.

features.png

Figura 15 : feature-urile unui film

### Profilul utilizatorului și profilul itemelor

În Figura 15 sunt prezentate caracteristicile luate în calcul la construirea profilului utilizatorului.Fiecare actor,director,tag în parte va fi reprezentat sub forma unui string cu un anumit weight[[7]](#footnote-8).

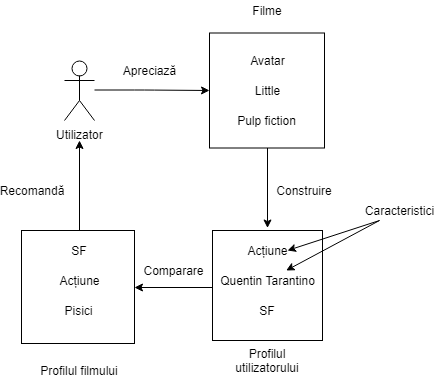


Figura 16 : arhitectura sistemului de recomandări bazat pe conținut

feature-weight.gif(15)

Relația 15 prezintă modul de calculare al ponderii pentru un anumit feature : suma ratingurilor care le-au primit filmele care conțin respectivul feature împărțită la numărul de filme.De observat este faptul că ratingurile au fost normalizate prin scăderea ratingului mediu acordat de utilizator.

Diferența dintre profilul utilizatorului și cel al filmelor constă în faptul că primul va conține un număr considerabil mai mare de feature-uri.De fiecare dată când un utilizator va nota un film ,profilului său i se vă adăuga orice feature prezent în profilul itemului,cu weight-ul corespunzător.Dacă în în profilul utilizatorului este prezent deja acel feature,îi va fi actualizată ponderea.

### Recomandările

Pe baza profilului construit al utilizatorului,recomandările se vor face comparându-l cu

toate celelalte profile ale itemelor existente și bineînțeles neevaluate încă de utilizator.

Va fi creat un scor de similaritate între profilul utilizatorului si profilul itemului.Pentru fiecare feature comun celor două profile(prezent atât în cel al utilizatorului cât și în cel al itemului) va fi adăugat la scorul similarității ponderea feature-ului respectiv.În contextul recomandării anumitor filme pentru un utilizator vor fi recomandate acele filme care au scorul similarității profilului cât mai mare.Pentru a putea fi luat în calcul un scor al similarității ,filmul ar trebui să aibe în comun cel puțin 1 + 5% [[8]](#footnote-9) feature-uri cu profilul utilizatorului.Condiția a fost pusă pentru a elimina filmele care au în comun maxim un feature.

## Abordarea hibridă

Până acum au fost prezentate 3 abordări diferite în alcătuirea unui sistem de recomandări.Luând în considerație descrierea fiecărui sistem în parte pot fi făcute următoarele afirmații:

-Sistemul de filtrare colaborativă User-User nu oferă garanția scalabilității,având în vedere că odată cu dezvoltarea aplicației numărul de utilizatori va fi mai mare decât numărul de filme.

-Sistemul de filtrarea colaborativă Item-Item este scalabil și oferă recomandări într-un timp optim.Depinde însă de activitatea comunității iar în contextul în care utilizatorul are gusturi unice nu promite livrarea de recomandări relevante.

-Sistemul de recomandări bazat pe conținut este independent de activitatea comunității și poate livra recomandări și utilizatorilor cu gusturi unice.Poate duce însă la supra-specializare,cu alte cuvinte nu elaborează recomandări în afara sferei profilului construit pentru utilizator.

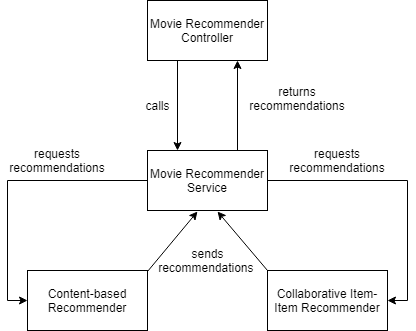


Figura 17 : abordare sistem hibrid

S-a renunțat la utilizarea sistemului de filtrare colaborativă User-User în detrimentul celui Item-Item care va folosi în implementare similaritatea calculată pe baza indexului Jaccard,similaritate pentru care a fost obținută cea mai mică rată a erorii.

Prin intermediul controller-ului din Figura 17 poate fi apelat serviciul de recomandări care poate returna recomandări din partea sistemului de recomandări bazat pe conținut , din partea celui bazat pe filtrare colaborativă Item-Item sau o combinație de recomandări rezultate de la cele două sisteme.În ultimul caz ,serviciul va transmite mai apoi recomandările rezultate după cum urmează :

- Prioritatea o au recomandările provenite de la sistemul Item-Item cu ratingul estimat peste 8.5 (pentru o scală de ratinguri naturale cu valori cuprinse în intervalul [1,10]).

- Apoi va fi preluată recomandarea cu cel mai bun scor al similarității cu profilul utilizatorului din partea sistemului bazat pe conținut.

- Sunt adăugate apoi toate recomandările cu un rating prezis de cel puțin 7.5 din partea sistemului de filtrare colaborativă.

-Se completează lista de recomandări cu filme recomandate de sistemul de recomandări bazat pe conținut.

S-a decis prioritizarea sistemului bazat pe filtrare colaborativă deoarece acesta a prezentat rezultate optime atât pentru un număr relativ mic de utilizatori cât și pentru unul mare.În eventualitatea în care nu pot fi prezise filme cu o valoare a ratingului cât mai mare,va interveni sistemul de recomandări bazat pe conținut.

# Arhitectura aplicației și tehnologii utilizate

## Arhitectura Aplicației

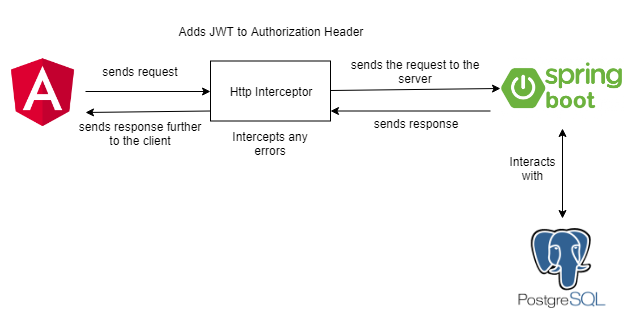


Figura 18 : diagramă arhitecturală

În figura prezentată mai sus este descris modul în care sunt procesate requesturile conținând JSON[[9]](#footnote-10)-uri. Http Request-urile clientului sunt interceptate de Http Interceptor-ul din Angular care le adaugă tokenul JWT[[10]](#footnote-11) în header.Request-urile sunt trimise mai depare apoi către server.Serverul interacționează cu baza de date de PostgreSQL și trimite răspunsul înapoi către client.Răspunsul de la server este interceptat tot de Http Interceptor care trimite după eventuala tratare a erorilor răspunsul mai departe către client.

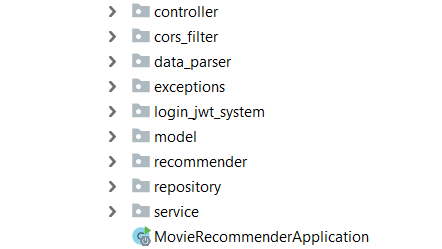


Figura 19 : structura Back-end

Figura 19 ilustrează modul de structurare a claselor pe partea de back-end.Pachetele controller,service,repository și model reprezintă componentele Spring Boot care facilitează tratarea request-urilor de la client precum și interacțiunea cu baza de date.Pachetul login\_jwt\_system conține componentului sistemului de login bazat pe JSON Web Token.Clasele din pachetul recommender reprezintă implementările sistemelor de recomandări prezentate plus clasa RecommenderTester care a efectuat testele pe baza acestora.

Pachetul data\_parser s-a ocupat cu parsarea datelor din CSV[[11]](#footnote-12)-uri și introducerea acestora în sistem.Cu alte cuvinte cu aceste date obținute a fost populată baza de date.Datasetul utilizat a fost TMDB 5000 Movies Dataset[[12]](#footnote-13) și a constat într-o suită de filme pentru care se cunoșteau : Actori,Genuri,Taguri și alte detalii semnificative care au fost utilizate cu scopul de a testa aplicația , sistemul de recomandări bazat pe conținut în principal dar și rezultatele sistemelor de recomandări în interacțiunea cu utilizatorul.



Figura 20 : structura componentelor Front-End

În Figura 20 este prezentată structura aplicației pe partea de Front-End.Componentele din pachetul services interacționează în mod direct cu partea de back-end ,făcând requesturi către server și trimițând răspunsul primit către componentele din pachetul pages.Acestea sunt utilizate pentru crearea efectivă a interfeței aplicației pe care utilizatorul o accesează.În folderul models sunt prezente descrierile obiectelor care mapează tabelele existente în baza de date.

## Tehnologii utilizate

### Spring Boot

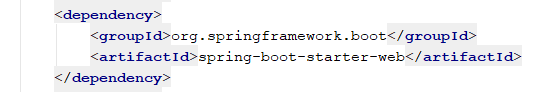


Figura 21 : spring boot starter dependency

Spring-ul reprezintă poate cel mai popular framework utilizat pentru crearea de aplicații web în limbajul de programare Java.Motivația alegerii Spring Boot-ului ca si framework vine din următoarele considerente :

* Tomcat,Jetty sau Undertow integrat
* Auto-configurare
* Nu necesită configurarea XML

### Spring Data JPA

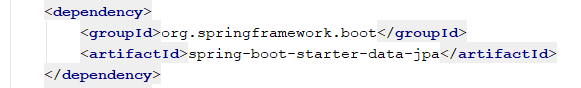


Figura 22 : spring data jpa dependency

Utilizarea Spring Data JPA aduce multe beneficii în cazul dezvoltării unei aplicații web iar cel mai important dintre ele este simplificarea codului utilizat în repository-ul care comunică cu baza de date.

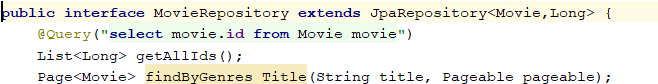


Figura 23 : spring data jpa repository

În Figura 23 poate fi observată modalitatea simplă de creare a Query-urilor personalizate prin utilizarea adnotarii @Query urmată de query-ul SQL.De asemenea se pot obține rezultate paginate prin utilizarea interfeței Pageable.

Query-urile sunt generate după numele metodelor,de exemplu o listă de filme paginate va fi returnată în funcție de titlul unui gen.

### Spring security

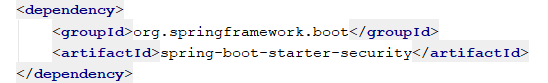


Figura 24 : spring security dependency

Spring security este utilizat pentru securizarea aplicațiilor bazate pe Spring.Acesta oferă suport pentru autentificare si autorizare,oferind protecție contra atacurilor care au ca țintă utilizatorii.

Spring security a fost integrat în aplicație pentru a facilita utilizarea JSON Web Token-urilor.

### Angular

Angular reprezintă un framework JavaScript bazat pe componente.Prin componente se înțelege împărțirea paginilor Web pe elemente care sunt independente de context,oferă propriul lor aspect și cel mai important sunt reutilizabile în orice pagină la orice moment.

angular_modules.png

Figura 25 : importarea modulelor în Angular

Un alt avantaj al utilizării Angular-ului este reprezentat de posibilitatea importării anumitor module care descriu anumite componente existente și pot fi integrate în aplicație cu ușurință(în cazul de față au fost preluate componente care oferă suport pentru paginare,ratinguri sau butoane) din frameworkul ng-bootstrap.

### PostgreSQL

PostgreSQL este un sistem de baze de date relaționale.Este gratuit și open-source și

asigură integritatea datelor,extensibilitatea și oferirea de soluții performante.

Modelarea bazei de date este realizată prin intermediul pgAdmin care oferă o platformă intuitivă și ușor de utilizat.

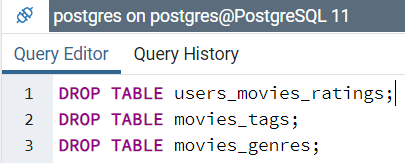


Figura 26 : query tool PostgreSQL

Pot fi create tabele de asemenea și prin intermediul query tool-ului prezent în care

se folosește de limbajul PL/pgSQL similar cu PL/SQL.

# Concluzii finale

Scopul aplicației a fost dezvoltarea unei aplicații web capabilă sa livreze recomandări pentru filme online.Dezvoltarea aplicaței a constat în următoarele etape:

* Configurarea aplicației web atât pe partea de front-end cât și pe partea de back-end.
* Studierea și analizarea tipurilor de sisteme de recomandări urmată de implementarea acestora și testarea pe anumite seturi de date.
* Analizarea rezultatelor testelor pentru conceperea unui sistem de recomandări hibrid care să aibă cât mai puține vulnerabilități.

Luând în calcul tema lucrării și scopurile ei inițiale se poate spune că rezultatul obțnut

este cel propus.

În eventualitatea continuării dezvoltării acestei aplicații există numeroase metode prin care aceasta poate fi extinsă :

* Adăugarea de comentarii/recenzii pentru filme și utilizarea unor metode de interpretare a limbajului natural(metoda tf-idf[[13]](#footnote-14) de exmplu) pentru adăugarea de noi feature-uri în sistemul de recomandări bazat pe conținut.
* Adăugarea de metode care să analizeze ratingurile implicite,cum ar fi timpul petrecut de utilizator citind descrierea unui film,filmele pe care utilizatorul le caută sau le citește descrierea.
* Căutarea filmelor după alte criterii în afară de titlu,nume actori,tag-uri etc.
* Trimiterea de notificări cu recomandări utilizatorului în momentul în care sunt găsite recomandări cu un scor foarte bun.

# Bibliografie

Ron Zacharski , A Programmer’s Guide to Data Mining

Stanford University , Lecture 42 – Content Based Recommendations

http://infolab.stanford.edu/~ullman/mmds/ch9.pdf

J. Ben Schafer ,Dan Frankowski,Jon Herlocker,Shilad Sen ,Collaborative Filtering Recommender Systems

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

https://mahout.apache.org/docs/0.13.0/api/docs/mahout-mr/org/apache/mahout/cf/taste/

https://www.slideshare.net/Cataldo/apache-mahout-tutorial-recommendation-20132014

https://www.javainuse.com/spring/boot-jwt

https://www.callicoder.com/spring-boot-jpa-hibernate-postgresql-restful-crud-api-example/

https://angular.io/start

https://docs.spring.io/spring-data/jpa/docs/current/reference/html/

1. Articole [↑](#footnote-ref-2)
2. https://grouplens.org/datasets/movielens/100k/ [↑](#footnote-ref-3)
3. Repetarea expermientului pe fiecare pereche set de antrenament-set de test și efectuarea mediei rezultatelor [↑](#footnote-ref-4)
4. https://grouplens.org/datasets/movielens/100k/ [↑](#footnote-ref-5)
5. <https://grouplens.org/datasets/movielens/10m/> [↑](#footnote-ref-6)
6. Proprieteate pe baza căreia se efectuează o predicție [↑](#footnote-ref-7)
7. Pondere în construirea profilului [↑](#footnote-ref-8)
8. 5% din numărul de feature-uri al filmului respectiv [↑](#footnote-ref-9)
9. https://www.json.org , JavaScript Object Notation [↑](#footnote-ref-10)
10. https://jwt.io/introduction/ , JSON Web Token [↑](#footnote-ref-11)
11. https://en.wikipedia.org/wiki/Comma-separated\_values , Comma separated values [↑](#footnote-ref-12)
12. https://www.kaggle.com/tmdb/tmdb-movie-metadata [↑](#footnote-ref-13)
13. <http://www.tfidf.com/> , term frequency – inverse document frequency [↑](#footnote-ref-14)