UNIVERZA V LJUBLJANI
FAKULTETA ZA ELEKTROTEHNIKO

Matevž Kunaver

Kombiniran vsebinsko-skupinski sistem za iskanje uporabniku prilagojenih multimedijskih vsebin

Doktorska disertacija

Mentor: prof. dr. Jurij Tasič
Somentor: prof. dr. Andrej Košir

Ljubljana, 2009

ZAHVALA

Na prvem mestu bi se rad zahvalil svojemu mentorju, prof. dr. Juriju Tasiču za vodenje pri doktorskem delu in za zagotovitev dobrih delovnih pogojev. Zahvala gre tudi somentorju prof. dr. Andreju Koširju, za pomoč pri matematičnih zapisih ter predvsem statistični analizi podatkov. Zahvala gre tudi drugim članom Laboratorija za digitalno obdelavo signalov, slik in videa za ustvarjalno in prijetno delovno vzdušje. Med njimi bi se še posebej rad zahvalil Tomažu Požrlu ter Matevžu Pogačniku, ki sta mi vedno priskočila na pomoč, ko je bilo potrebno izvesti kak poskus ali pa kaj predebatirati.

Za finančno podporo se zahvaljujem Ministrstvu za šolstvo, znanost in šport Republike Slovenije.

Za podporo, vzpodbudne besede in veliko mero potrpljenja v času doktorskega študija se zahvaljujem tudi svojima staršema, brez katerih bi mi danes gotovo še veliko manjkalo do zaključka podiplomskega študija.

Nenazadnje pa se moram zahvaliti tudi Danici, ki je proti koncu doktorata prinesla nov zagon v moje življenje, mi pomagala najti energijo za zadnje napore ter mi vedno stala ob strani z nasmehom na ustih.

Povzetek

V pričujoči disertaciji predstavljamo razvoj dveh sistemov za uporabniku prilagojeno iskanje vsebin. Personalizacija oziroma njeno podpodročje, uporabniku prilagojeno iskanje vsebin je postopek, pri katerem iz množice dostopnih vsebin izberemo tiste, za katere menimo, da jih bo uporabnik najbolj pozitivno sprejel (ocenil z visoko oceno). Zato potrebujemo podatke o uporabniku, ki jih dobimo s pomočjo analize njegovih preteklih dejanj – do katerih vsebin je dostopal in do katerih ne. Podatke, pridobljene s to analizo, nato shranimo v podatkovno strukturo, ki jo imenujemo uporabniški profil.

Testiranje smo izvajali na dveh zelo različnih podatkovnih množicah, kar nam je omogočilo testiranje sistema za iskanje primernih vsebin v dveh popolnoma različnih scenarijih. Prva podatkovna množica je bila dokaj prazna (vsebovala je samo majhen procent vseh možnih ocen) ter je tako omogočila simuliranje uspešnosti sistema v okolju, kjer se je sistem šele začel uporabljati (torej se uporabniki prvič prijavljajo in podajajo ocene samo za ozek nabor vsebin). Druga podatkovna množica pa je bila skoraj polna (vnesene so bile skoraj vse ocene) in je tako omogočila simuliranje uspešnosti sistema v okolju, kjer se sistem uporablja že dalj časa in so uporabniki ocenili že večino vsebin ali pa okolju, kjer se zelo poredko dodajajo nove vsebine.

Sistem smo preizkušali v dveh korakih. V prvem koraku smo z izbrano metodo zgradili uporabniške profile ter vanje shranili potrebne podatke. V drugem koraku pa smo nato s pomočjo teh profilov poiskali primerne vsebine za vsakega obstoječega uporabnika. Uspešnost sistema smo nato merili na podlagi primerjave izračunanih napovedanih ocen (torej vrednosti, s katero sistem ovrednoti, kako je vsebina primerna za uporabnika) ter dejanskih ocen, shranjenih v podatkovnih množicah. Na podlagi te primerjave smo izračunali vrednost mere F, ki se na področju uporabniku prilagojenega iskanja vsebin najbolj pogosto pojavlja kot mera uspešnosti sistema.

Pri razvoju prvega sistema za iskanje vsebin smo se osredotočili na področje skupinskega filtriranja, kjer je glavni cilj iskanje uporabnikov s podobnim okusom. V ta namen smo testirali dve različni metodi iskanja uporabnikov, da bi ugotovili, katera so bolje obnese (torej katera metoda uporabniku bolj zanesljivo predlaga vsebine). Ugotovili smo, da se najbolje obnese kombinacija obeh metod, ki tako deluje na širšem naboru uporabnikov. Pri testiranju metod za izračun napovedane ocene (torej metod, s pomočjo katerih iščemo primerne vsebine na podlagi ocen podanih s strani najbližjih sosedov) smo preizkusili metodo uteženе vsote ter metodo Bayesovega približka. Na podlagi rezultatov smo razvili novo metodo, osnovano na uteženi vsoti, ter jo uporabili v prvem sistemu.

V prvem sistemu smo tako uporabili več metod s področja skupinskega filtriranja, da smo dosegli najboljše rezultate. Poleg same izbire metod pa je bilo pomembno tudi vprašanje nastavitve parametrov teh metod. Zato smo vse metode testirali pri različnih vrednostih parametrov ter na različnih podatkovnih množicah. Prišli smo do pomembnega spoznanja, da optimalna vrednost parametrov (torej vrednost parametrov, pri kateri sistem najbolje deluje in išče primerne vsebine z največjo zanesljivostjo) ne obstaja. Parametri so namreč odvisni od lastnosti podatkovne množice, na kateri se metoda uporablja. Zato je potrebno slediti rezultatom sistema in v primeru ko ugotovimo, da so rezultati padli pod določeno mejo, poiskati nove vrednosti parametrov, pri katerih bo sistem optimalno deloval.

Tako razvit sistem je vračal zelo dobre rezultate, vendar še ni primeren za rabo v realnih sistemih. Težava, na katero smo naleteli, je računska zahtevnost, zaradi katere je sistem za iskanje primernih vsebin za posameznega uporabnika potreboval do 6 minut, kar ni

sprejemljivo. V ta namen smo razvili nov, kombiniran sistem, v katerem smo prvotnemu sistemu dodali metode iz področja vsebinskega filtriranja vsebin.

Obstoječe uporabniške modele smo razširili ter jim dodali žanrske preference, ki so nam predstavljale, kako uporabnik sprejema posamezno vrednost žanra. Novost, ki smo jo uvedli, je iskanje najbližjih sosedov na podlagi teh preferenc. Klasični sistemi skupinskega filtriranja primerjajo dva uporabnika na podlagi ocen vseh vsebin, ki sta jih oba ocenila. Število ocen, v katerih se uporabnika prekrivata, s časom narašča in tako raste tudi število računskih operacij, ki jih mora sistem izvesti, da uporabnika primerja. Z našim sistemom pa sistem dva uporabnika vedno primerja glede na fiksno število obstoječih preferenc (preferenc je toliko, kolikor je različnih žanrov, le-ti pa so bolj ali manj fiksno določeni).

Tako dobljeni sistem je deloval opazno hitreje (po par sekund na uporabnika) in z večjo zanesljivostjo kot prvotno razviti sistem. S tem smo tudi pokazali, da kombinirani sistemi delujejo bolje kot sistemi, osnovani samo na skupinskem ali samo na vsebinskem principu filtriranja vsebin, kar je razvidno iz primerjave potekov mer F posameznih sistemov.

Ključne besede:

Uporabniku prilagojeno iskanje vsebin, skupinsko filtriranje vsebin, vsebinsko filtiranje vsebin, kombinirani sistemi.

Abstract

The following Ph.D. thesis presents the development of two different user modeling systems. User modeling (or recommender system) is a procedure used to filter available content in order to present the user with a selection of items which are deemed to be interesting for this particular user. In order to achieve this, the system requires information about the user. This information is obtained by analyzing user history (history of user's previous interaction with the system). Result from this analysis are then stored in a special data structure called 'User Model'.

We have tested our recommender systems with two very different datasets. This enabled us to create two very different testing environments (scenarios). The first dataset was almost empty (it included only a small percent of all possible ratings) and was therefore useful for creating an early-stage scenario, where we simulated users who have only recently begun to use the system. The other dataset was almost full (almost all possible ratings were inputted) and enabled us to simulate a scenario where the users were using the system for a longer period of time.

We tested the system in two stages. In the first stage we created user models and updated them with the necessary infromation. In the other stage we then used this models to recommend items for each of our users. We measured the efficiency of our system by calculating a predicted rating for each item and then comparing it with the user's actual rating. Based on this comparison we calculated the F-measure, which is used frequently in the field of user modeling.

During the development of our first system we concentrated on collaborative recommenders that are based on nearest neighbor search. We tested two different metrics for nearest neighbor selection, to determine which results in better performance. Our results have shown that a combination of both methods works best. When testing methods for calculating predicted ratings, we tested Pearson's weighted sum and True Bayesian Estimate. Based on our results we developed a new method – adjusted weighted sum.

Our first system included several collaborative methods to improve performance. But just selecting these methods was insufficient. We had to determine the optimum parameter values for each method. In order to determine this we tested the methods with several different parameter values and using different datasets. The results have shown that there are no fixed optimum values for these parameters. Rather the parameters are dependant on the characteristics of the dataset used by the recommender system. Therefore we must track the statistics of the dataset and adjust parameters accordingly.

The first recommender system we developed was performing efficiently (with high values of F-measure), but had a serious problem. The system required a lot of time (up to 6 minutes) to create a list of recommendations for a single user. In order to correct this we have developed a new, hybrid recommender.

We expanded existing user profiles by adding genre preferences. These preferences reflected the user's preferences for each genre type. Our contribution was using these preferences to select nearest neighbors. A 'classical' collaborative recommender selects neighbors based on item-by-item rating comparison. The number of items in which two users overlap (meaning that they both rated the same item), grows with time and therefore the system must also perform a lot more operations to select appropriate neighbors. Using our approach, the number of comparisons is constant as the number of possible genres does not change.

Our system was working noticeably faster while still maintaining a high level of efficiency (high values of F-measure). This also demonstrates that hybrid recommenders perform better due to combination of different approaches.

Keywords: item recommendations, collaborative recommender, content-based recommender, hybrid recommender.

Kazalo

1. UVOD ... 1

1.1 Uporabniku prilagojeno iskanje digitalnih vsebin ... 2

1.2 Razvoj osebnih digitalnih naprav in komunikacijskih povezav ... 3

1.3 Opisovanje (multimedijskih) vsebin ... 4

1.3.1 Metapodatki ... 4

1.3.2 Metapodatkovni standardi ... 5

1.4 Uporabniški model ... 12

1.5 Struktura dela ... 12

2. UPORABNIKU PRILAGOJENO ISKANJE VSEBIN ... 13

2.1 Splošen pregled področja in osnovnih pojmov ... 13

2.1.1 Uporabniški model ... 14

2.1.2 Učenje uporabniškega modela ... 14

2.1.3 Izbira primernih vsebin za uporabnika ... 14

2.1.4 Zbiranje povratnih informacij ... 15

2.2 Terminologija in matematične oznake ... 16

2.3 Področja uporabniku prilagojenih storitev ... 18

2.3.1 Interaktivna digitalna TV ... 18

2.3.2 Spletno hranjenje multimedijskih vsebin ... 19

2.3.3 Reklamna sporočila ... 19

2.3.4 Trgovine ... 19

2.4 Postopki in metode za ocenjevanje primernosti vsebin ter njihovo razvrščanje ... 20

2.4.1 Vsebinsko filtriranje ... 20

2.4.2 Skupinsko filtriranje ... 22

2.4.3 Kombinirani pristopi ... 26

2.4.4 Ostale skupine postopkov za filtriranje vsebin ... 27

2.5 Pomankljivosti na področju uporabniku prilagojenega iskanja vsebin ... 27

2.5.1 Redkost podatkov ... 27

2.5.2 Dodajanje novih uporabnikov ... 27

2.5.3 Vpeljava novih vsebin ... 28

2.5.4 Prekomerno prileganje ... 28

2.5.5 Pomanjkanje standardov ... 28

3. TESTNO OKOLJE ... 31

3.1 Testno okolje ... 31

3.2 odatkovne množice ... 32

3.2.1 EachMovie ... 32

3.2.2 Siol ... 34

3.3 Postopek testiranja ... 36

3.4 Mera učinkovitosti ... 37
3.4.1 Delež najdenih, natančnost in mera F ... 37
3.4.2 Statistično testiranje ... 39
3.5 Predstavitev rezultatov ... 40

4. RAZVOJ SISTEMA ZA SKUPINSKO FILTRIRANJE ... 41
4.1 Potek skupinskega modeliranja ... 41
4.1.1 Gradnja uporabniškega modela ... 41
4.1.2 Iskanje primernih vsebin ... 43
4.2 Zasnova sistema ... 45
4.3 Izbira metode iskanja sosedov ... 46
4.3.1 Pearsonov korelacijski koeficient ... 50
4.3.2 Evklidova razdalja ... 51
4.3.3 Kombinacija obeh metod ... 52
4.4 Izračun napovedane ocene ... 53
4.4.1 Utežena vsota ... 54
4.4.2 Bayesov približek ... 55
4.4.3 Prilagojena utežena vsota ... 56
4.5 Rezultati ... 59
4.5.1 Primerjava metod za iskanje primernih vsebin ... 59
4.5.2 Čas procesiranja ocen uporabnikov ... 60
4.6 Možnosti za nadaljnji razvoj ... 61
4.6.1 Modularni sistem ... 61

5. IZBIRA VREDNOSTI PARAMETROV SISTEMA ZA SKUPINSKO ISKANJE PRIMERNIH VSEBIN ... 65
5.1 Izbira pravilnega števila sosedov ... 65
5.1.1 Eachmovie ... 66
5.1.2 Siol ... 67
5.1.3 Dokončna izbira optimalnega števila ... 68
5.2 Izbira uteži za Bayesov približek ... 69

6. KOMBINIRAN VSEBINSKO-SKUPINSKI PRISTOP ... 73
6.1 Osnovna ideja predlaganega pristopa k uporabniškemu modeliranju ... 73
6.2 Eigentaste ... 73
6.3 Razvoj novega pristopa za iskanje najbližjih sosedov ... 75
6.4 Gradnja vsebinskega profila ... 77
6.5 Izbira primernih sosedov ... 78
6.6 Iskanje primernih vsebin ... 78
6.7 Rezultati kombiniranega sistema ... 79
6.7.1 Računska zahtevnost ... 80

7. ZAKLJUČEK ... 81

7.1 Izvirni prispevki k znanosti 84

Kazalo slik

Slika 1:Sistem za uporabniku prilagojeno iskanje vsebin *1*
Slika 2: Potek uporabniku prilagojenega iskanja vsebin *3*
Slika 3: Primer hitre rasti dostopnih vsebin (wikipedia) *4*
Slika 4: potek UM (učenje, model, recomm) *13*
Slika 5: Personaliziran programski vodič *18*
Slika 6: Spletni vmesnik trgovine amazon.com s primerom predlaganih vsebin *20*
Slika 7: Vsebinsko filtriranje *21*
Slika 8: Skupinsko filtriranje *22*
Slika. 9: Kombinirani pristopi *26*
Slika 10: E-R diagram EachMovie podatkovne baze *33*
Slika 11: E-R diagram Siol podatkovno bazo *35*
Slika 12: Primer 10-kratnega prečnega preverjanja *37*
Slika 13: Primer podjanja rezultatov *40*
Slika 14: Primerjava uporabnikov *42*
Slika 15: izbira primernih vsebin *44*
Slika 16: Tok izvajanja sistema za skupinsko filtriranje vsebin *45*
Slika 17: Primer vsebine uporabniškega modela *46*
Slika 18: Rezultati prilagojene utežene vsote ter Pearsonovega korelacijskega koeficienta *50*
Slika 19: Rezultati prilagojene utežene vsote ter Evklidove razdalje *52*
Slika 20: Preklop distanc *53*
Slika 21: Potek mere F pri uteženi vsoti *54*
Slika 22:Potek mere F pri Bayesu *56*
Slika 23: Potek mere F pri prilagojeni uteženi vsoti *58*
Slika 24: Primerjava metod – Evklidova razdalja *59*
Slika 25: Primerjava metod – Pearsonov korelacijski koeficient *59*
Slika 26: Časovna zahtevnost v odvisnosti od števila vnesenih ocen uporabnika *60*
Slika 27: Delitev skupinskega filtriranja na module *63*
Slika 28: Vpliv števila sosedov – Pearsonov korelacijski koeficient - EachMovie *66*
Slika 29: Vpliv števila sosedov – Evklidova razdalja - EachMovie *66*
Slika 30: Vpliv števila sosedov – Pearsonov korelacijski koeficient - SIOL *67*
Slika 31: Vpliv števila sosedov – Evklidova razdalja - SIOL *68*
Slika 32: Uteži bayesovega približka - SIOL *70*
Slika 33: Uteži Bayesovega približka - EachMovie *70*
Slika 34: Porazdelitev uporabnikov z uporabo Eigentaste algoritma v originalnem članku *73*
Slika 35: Porazdelitev uporabnikov z uporabo Eigentaste algoritma na naši podatkovni množici *74*
Slika 36: Arhitektura kombiniranega vsebinsko-skupinskega sistema *76*
Slika 37: Primerjava obeh sistemov *79*

Kazalo tabel

Tabela 1: Opis vsebine z atributi *16*
Tabela 2: SIol EPG *34*
Tabela 3: Dodeljevanje v matriko razvrščanja *38*
Tabela 4: Vrednosti atributa žanra a_i *77*

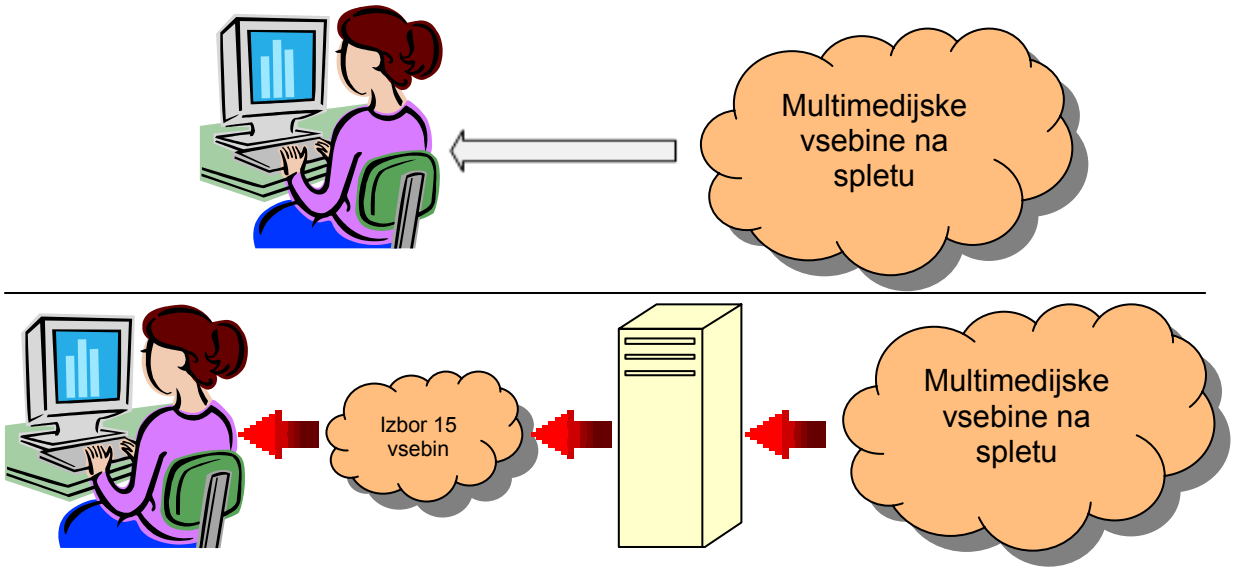
1. Uvod

Z razvojem digitalnih naprav ter vedno večjo razširjenostjo spleta je uporabniku na voljo vedno večja količina vsebin. Poleg tega je uporabniku omogočen vedno lažji dostop do teh vsebin, saj postajajo prenosne poti enostavnejše, bolj razširjene ter predvsem, dostopnejše. Uporabnik se je sedaj iz pasivnega sprejemnika vsebin spremenil v aktivnega uporabnika, ki sam izbira vsebine, ki jih želi uporabljati. Z razvojem novih naprav, kot so digitalni videorekorderji, mobilni terminali (telefoni in dlančniki) ter nenazadnje tudi osebni in prenosni računalniki, lahko uporabnik do vsebin dostopa kjerkoli in kadarkoli.

Te spremembe kateremukoli uporabniku omogočijo dostop do velike množice raznolikih vsebin. Posledica tega pa je tudi to, da je uporabnik soočen s skoraj nepregledno količino vsebin, zato sam težko izbere primerne oziroma zanimive vsebine v doglednem času. Iskanje je zato postalo naporno in uporabniku neprijazno. Dodatna težava, na katero uporabnik naleti, je samo opisovanje vsebin, ki je nestrukturirano, nepopolno ter osnovano na mnogo različnih standardih. Podatke, s pomočjo katerih opisujemo vsebine, imenujemo metapodatki.

Za pomoč uporabniku pri iskanju primernih vsebin se je začel razvoj na področju uporabniku prilagojene izbire vsebin (ang. *recommender systems*). Osnovni princip uporabniku prilagojenega iskanja vsebin je opazovanje uporabnika pri interakciji z vsebinami ter njihova analiza, na podlagi katere sistem nato zgradi uporabniški model (ang. *user model*). S pomočjo uporabniškega modela sistem nato preuči ostale potencialne vsebine in med njimi izbere tiste, za katere misli, da bodo s strani uporabnika najbolj pozitivno sprejete. Ker se algoritmi za tako iskanje vsebin izvajajo na digitalnih napravah, ki lahko v zelo kratkem času preučijo veliko večjo količino vsebin kot bi jih uporabnik sam.

Cilj sistema za uporabniku prilagojeno iskanje vsebin je torej da uporabniku olajša iskanje vsebin tako, da mu ponudi manjši izbor vsebin, kot je prikazano na sliki 1.



Slika 1:Sistem za uporabniku prilagojeno iskanje vsebin

Cilj pričujoče disertacije je izdelava sistema za uporabniku prilagojeno iskanje vsebin, osnovanega na pristopu skupinskega filtriranja, ki bo uporabniku kvalitetno pomagal poiskati primerne vsebine. Sistem bo zasnovan za uporabo v resničnih razmerah, torej da bo deloval hitro ter zanesljivo tudi v okolju z večjim številom vsebin ter uporabnikov. Sistem bomo testirali na dveh obstoječih podatkovnih množicah. Podatkovni množici vsebujeta zgodovino ocenjevanja večjega števila uporabnikov. Na podlagi te zgodovine bomo z našimi pristopi sestavili uporabniške profile ter z njihovo pomočjo uporabnikom skušali poiskati primerne vsebine. S primerjavo resničnih ocen ter naših izračunanih napovedanih ocen bomo ocenili uspešnost sistema ter na podlagi teh rezultatov ugotovili, kakšne prilagoditve so še potrebne v sistemu. Prva verzija sistema bo zasnovana na čisto skupinskem pristopu. Po tej fazi testiranja bo sistem nadgrajen z vpeljavo vsebinskih pristopov z namenom optimizacije procesa iskanja primernih vsebin ter uvedbo interesnih skupin. Nadgrajen sistem bomo nato preizkusili v istih pogojih z enakimi podatkovnimi množicami. Dobljene rezultate bomo nato primerjali s prvotnim sistemom ter ocenili, kateri sistem se v danih okoliščinah bolje obnese.

1.1 Uporabniku prilagojeno iskanje digitalnih vsebin

Prvi primer uporabniku prilagojenega iskanja vsebin lahko zasledimo v osemdesetih letih 20. stoletja v članku, ki ga je objavil Salton [1]. V njem je predstavil tehniko za iskanje tekstovnih dokumentov s pomočjo vektorjev besed. V nadaljnjih letih razvoja se je področje uporabniku prilagojenega iskanja vsebin razširilo in sedaj pokriva širok spekter storitev. Primeri takih storitev so personalizirano iskanje spletnih dokumentov [2, 3, 4, 5], iskanje sporočil v forumskih skupinah [6, 7, 8], filtriranje elektronske pošte [9], iskanje multimedijskih vsebin ter uporabniku prilagojeni oglasi [10, 11, 12, 13, 14].

Metode personaliziranega iskanja vsebin se med seboj razlikujejo glede:

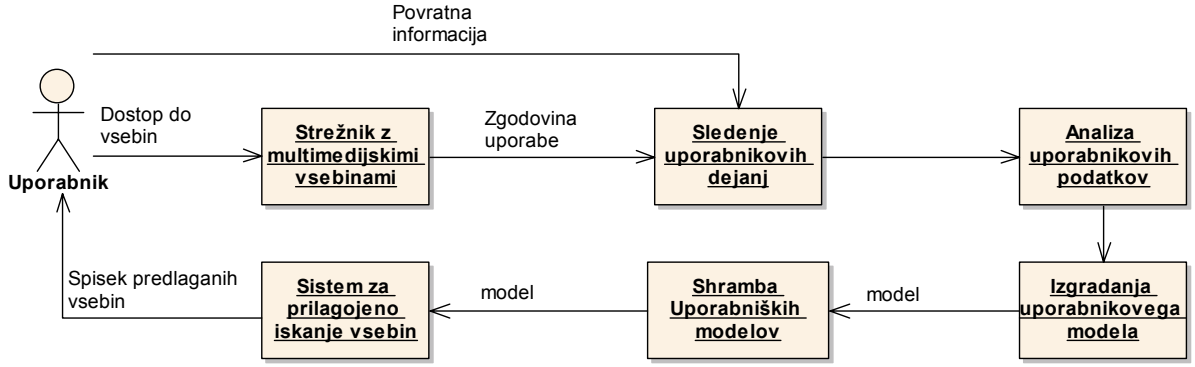
- Uporabljenih algoritmov in podatkovne strukture.
- Postopekov iskanja primernih vsebin.
- Oblike povratne informacije o uporabnikovem mnenju o ponujeni vsebini.
- Vrsta in število opisnih atributov, uporabljenih pri iskanju vsebin.

Na področju personalizacije je razvitih veliko različnih algoritmov, ki se lahko v isti situaciji popolnoma drugače odzovejo. Zato je v razvoju sistema za prilagojeno iskanje vsebin najprej potrebno preučiti razmere, v katerih se bodo sistemi uporabljali, ter na podlagi tega izbrati primernen algoritem. Nekateri izmed bolj razširjenih algoritmov so na primer vektorji besed [15, 16], odločitvena drevesa [14], (naivni) Bayesov klasifikator [14, 16], Bayesove verjetnostne mreže [13], metode k-najbližjih sosedov [10], metode podpornih vektorjev [17, 18] (ang. Support vector machines) ter druge.

Prav tako se sistemi razlikujejo po pristopu do iskanja primernih vsebin. Bolj kot sama izbira algoritma je tu pomembna izbira splošnega pristopa, na katerem je osnovano iskanje vsebin. Če primerne vsebine iščemo na podlagi opisov vsebin, ki jih je trenutni uporabnik že ocenil, govorimo o vsebinskem pristopu [2] (ang. Content-based). Če se vsebine iščejo na podlagi iskanja uporabnikov s podobnim okusom, pa govorimo o skupinskem pristopu [2] (ang. Collaborative-based).

Večina sistemov uporabniku prilagojenega iskanja vsebin ne morejo delovati brez povratne informacije o tem, kakšno je uporabnikovo mnenje o ponujenih vsebinah. Brez te povratne informacije sistem namreč ni sposoben ugotoviti uporabnikovih želja ter posledično zgraditi uporabniškega modela.

Tipični scenarij uporabniku prilagojenega iskanja vsebin je prikazan na sliki 2:



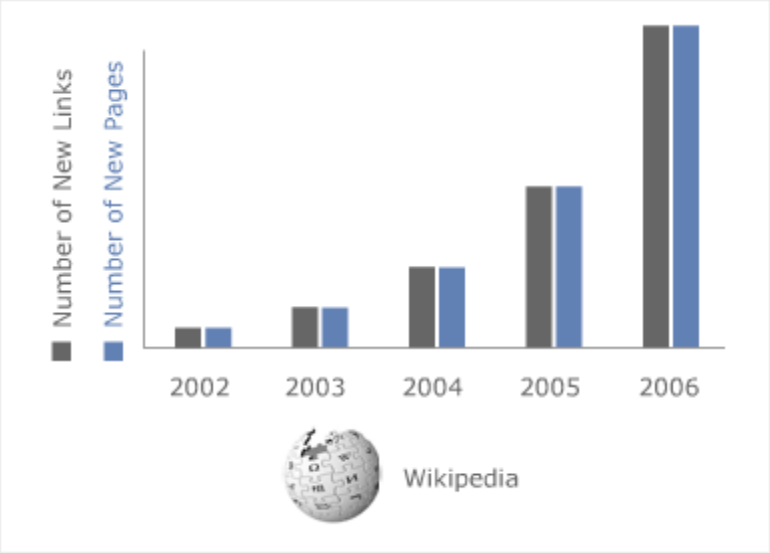
Slika 2: Potek uporabniku prilagojenega iskanja vsebin

Uporabnik preko strežnika z multimedijskimi vsebinami dostopa do vsebine, ki ga trenutno zanima. Sistem lahko sledi uporabnikovim dejanjem na dva načina – implicitno preko zgodovine uporabe, ki jo strežnik sporoča sistemu za sledenje ali pa eksplicitno, ko uporabnik direktno pove svoje mnenje o vsebini, do katere trenutno dostopa (več o tem v poglavju 2.1.4). Sistem nato zbrane informacije posreduje v modul za analizo, kjer algoritem za uporabniku prilagojeno iskanje vsebin iz teh podatkov izlušči podatke, ki jih potrebuje za iskanje vsebin. Potrebni podatki se razlikujejo od sistema do sistema. S pomočjo teh podatkov nato zgradi uporabnikov model, ki vsebuje povzetek uporabnikove zgodovine uporabe ter nastavitve za iskanje novih vsebin. Uporabnikov model se hrani v posebni podatkovni bazi, saj ni smiselno, da bi se gradil vsakič, ko se uporabnik prijavi v sistem. Ko se uporabnik prijavi v sistem (ali pa ko sistem opravlja posodabljanje vseh vsebin), se njegov profil naloži iz podatkovne baze v sistem za prilagojeno iskanje vsebin. Le-ta nato na podlagi podatkov v uporabnikovem profilu na strežniku vsebin poišče vsebine, za katere oceni (izračuna), da bi uporabnika najbolj zanimale. Te vsebine nato predlaga uporabniku, najbolj pogosto v obliki spiska predlogov, urejenega po primernosti.

1.2 Razvoj osebnih digitalnih naprav in komunikacijskih povezav

Hiter razvoj in prodor osebnih komunikacijskih naprav ter osebnih digitalnih naprav je opazno vplival na količino vsebin ter informacij, ki so v vsakem hipu dostopne uporabniku. S širitvijo spleta ima uporabnik dostop do vsebin kjerkoli ter kadarkoli. Zato je uporabniku prilagojeno iskanje vsebin vedno bolj potrebno.

Z razvojem prenosnih komunikacijskih naprav (GSM telefonov [19], ki so sedaj postali že prenosne multimedijske naprave), ima uporabnik omogočen dostop do spleta, strujanja vsebin ter celo kreiranja lastnih vsebin z vgrajenimi zajemalnimi napravami (kamera, fotoaparat). Močno je napredovala tudi digitalna televizija, ki sedaj v kombinaciji z digitalnimi osebnimi videorekorderji predstavlja novo interaktivno platformo, s pomočjo katere uporabnik dobi dostop do multimedijskih vsebin.



Slika 3: Primer hitre rasti dostopnih vsebin (wikipedia)

Uporabnik je torej danes skoraj povsod povezan preko spleta in lahko dostopa do multimedijskih vsebin. Zaradi vedno večje preprostosti uporabe osebnih digitalnih naprav strmo narašča tudi količina vsebin, ki si jih uporabniki izmenjujejo med seboj (npr youtube). Uporabnik ja zato zasut z veliko količino vsebin, med katerimi mora sam najti tisto, ki mu ustreza. Vedno večja razširjenost omrežij tipa vsak z vsakim (ang. *peer-to-peer*), v katera je vsak uporabnik vključen kot ponudnik in hkrati tudi kot odjemalec vsebin, še dodatno poveča količino dostopnih vsebin in tako vpliva na problematiko identifikacije uporabniku zanimivih vsebin.

1.3 Opisovanje (multimedijskih) vsebin

S širjenjem spleta ter spletnih vsebin je postalo opisovanje teh vsebin vedno bolj problematično. Spletne vsebine so bile na začetku opisane skladno s standardom HTML (ang. HyperText Markup Language), ki je namenjen predvsem oblikovanju vizualne podobe spletne strani vsebine, nudi pa zelo malo možnosti za opisovanje same vsebine spletne strani. Zato se je razvila vrsta novih pristopov za opisovanje multimedijskih vsebin, med katerimi najbolj izstopajo Dublin Core [20], MPEG-7 [21] in TV Anytime [22]. Dublin Core, MPEG-7 ter TV Anytime so namenjeni opisovanju vsebin s stališča dogodkov, objektov, oseb, abstraktnih konceptov, igralcev, avtorjev in podobno.

1.3.1 Metapodatki

Metapodatki [23] so podatki o podatkih, torej podatki, s katerimi opisujemo oz. dodatno opredeljujemo druge podatke. Izraz meta izhaja iz grškega jezika in pomeni nekaj višjega, bolj osnovnega. Metapodatki o nekem določenem podatku (ta je lahko čisto abstrakten pojem ali pa nek konkreten fizičen predmet) torej predstavljajo dodatno informacijo, s katero lahko o tem podatku izvemo bistveno več kot pa bi v primeru, če takšnih metapodatkov ne bi poznali.

Namen uporabe metapodatkov je večplasten. Poleg same opisnosti, s katero podatkom dodamo »vrednost« in s tem omogočimo bolj konstruktivno uporabo podatkov, so metapodatki zelo uporabni v primerih, ko želimo v množici podatkov poiskati tistega, ki nas

zanima. To lahko poteka tako v »analogni« obliki (ročno iskanje po katalogih) ali pa v avtomatizirani obliki s pomočjo računalnikov. Metapodatki se v tem primeru uporabijo kot iskalni pojmi, s katerimi iskalni mehanizmi preletijo nek seznam, katalog oz. druge vrste bazo in iščejo podobnost z vnesenimi podatki.

Metapodatki obstajajo v zelo različnih oblikah in načinih zapisa. Lahko so pripeti k samim dokumentom in so sestavni del le-teh ali pa jih shranimo ločeno in zato tudi iskanje po njih poteka ločeno od samih podatkov (katalogi, knjižnice ...). V obeh primerih pa so lahko precej različno strukturirani. Najbolj enostavna oblika so t.i. ključne besede, kar pomeni, da so metapodatki podani kot zaporedje besed in besednih zvez, ki označujejo določen podatek. Lahko pa jih zapišemo bolj strukturirano, pri čemer se običajno uporabljajo dogovorjena opisna polja. Tako na primer pri opisu knjige v ustrezna polja zapišemo naslov, avtorja, letnico izdaje, ključne pojme itd. Tak način zapisa omogoča že določen nivo semantike, kar nam lahko precej pomaga pri iskanju po podatkovni bazi.

Z metapodatki lahko opremimo tako rekoč vsako fizično entiteto, ki predstavlja podatek oz. informacijo ali pa je le vir informacij. Pri tem gre lahko za dejansko fizično predstavitev informacije (knjige, revije, katalogi, slike ...) ali pa le za abstraktno predstavitev informacije (tekstovne, slikovne, video, avdio in druge datoteke v računalniški obliki).

Primer zapisa metapodatkov :

```
<META NAME="DC.Date" CONTENT="(SCHEME=ISO8601) 1998-01-16">

<META NAME="DC.Title" CONTENT="User Guidelines for Dublin Core creation (Nordic Metadata Project)">

<META NAME="DC.Creator" CONTENT="Hansen, Preben">

<META NAME="DC.Creator.Address" CONTENT="preben@sics.se">

<META NAME="DC.Subject.keyword" CONTENT="Dublin Core, Metadata, User Guidelines"> <META NAME="DC.Type" CONTENT="User Guide, Tutorial">

<META NAME="DC.Identifier" CONTENT="(SCHEME=URL) http://www.sics.se/~preben/DC/ DC_guide.html">

<META NAME="DC.Language" CONTENT="(SCHEME=ISO.639-1) sv">
```

Zgornji zgled prikazuje zapis metapodatkov za publikacijo *»User guidelines for Dublin Core creation«*. Opisani so datum izdaje publikacije, naslov, avtor, avtorjev elektronski naslov, ključne besede vsebine publikacije, identifikator (v tem primeru je to lokacija publikacije na svetovnem spletu) ter jezik, v katerem je publikacija napisana. Vse te dodatne podatke (metapodatke) bi lahko uporabili pri iskanju te publikacije znotraj nekega kataloga oz. seznama.

1.3.2 Metapodatkovni standardi

Metapodatki so običajno sestavljeni iz nabora atributov oz. elementov, ki so potrebni za opis obravnavanega vira. Za primer vzemimo metapodatkovni sistem, ki se uporablja v knjižnicah – knjižnični katalog. Ta vsebuje nabor metapodatkovnih zapisov, ki opisujejo neko knjigo oz. drugo knjižnično enoto: avtorja, naslov, datum nastanka oz. objave, temo vsebine in identifikacijsko številko, ki označuje, kje točno na policah je knjiga postavljena.

Čeprav so se metapodatki uporabljali že pred razcvetom interneta oz. svetovnega spleta, pa se je večje zanimanje za metapodatkovne standarde pojavilo šele z razvojem elektronskega založništva in digitalnih knjižnic. Na spletu se je pojavila ogromna množica informacij v digitalni obliki, ki jih je bilo težko razlikovati in jih koristno uporabiti. Iskanje informacij z uporabo ene izmed iskalnih spletnih strani je lahko zelo mučno, ko nam iskanje vrne na stotine ali kar tisoče potencialno zanimivih zadetkov. Standardizirani opisni metapodatki v tem kontekstu služijo za boljše iskanje relevantnih virov, predvsem v smislu iskanja po poljih (avtor, naslov ...).

V zadnjih letih je nastalo veliko metapodatkovnih standardov oz. zapisov, vsako znanstveno področje oz. disciplina ima vsaj nekaj takšnih predpisov, v naslednjih dveh podpoglavjih pa si bomo nekoliko bolj podrobno pogledali dva primera tovrstnih zapisov. Pri tem bomo nakazali, kaj vse omogočajo različni metapodatkovni zapisi, predvsem z vidika primerjave s pravimi ontološkimi zapisi, ki bodo opisani v kasnejših poglavjih. Prvi opisani metapodatkovni zapis je Dublin Core [20], ki se je uveljavil kot eden izmed osrednjih splošnih standardov za opis spletnih virov, drugi pa je TV Anytime [22], pobuda skupine mednarodnih organizacij in podjetij, katere del je tudi metapodatkovni zapis za opisovanje avdio-vizualnih virov.

1.3.2.1 Metapodatkovni standard Dublin Core

Dublin Core [20] je metapodatkovni standard, ki je namenjen opisovanju širokega spektra omrežnih virov. Standard vsebuje dva nivoja, preprostega (Simple) in kvalificiranega (Qualified). Dublin Core je preprost, a učinkovit nabor opisnih elementov. Preprost Dublin Core je sestavljen iz 15 elementov, kvalificirana različica Dublin Cora pa ima zraven še šestnajsti element »Audience« ter skupino kvalifikatorjev, pojmov, ki dodatno opredeljujejo semantiko opisnih elementov v tem smislu, da ti postanejo še bolj uporabni za odkrivanje oz. iskanje virov. Semantiko standarda Dublin Core je pomagala zgraditi mednarodna interdisciplinarna skupina strokovnjakov s področja knjižničarstva, računalniških znanosti, kodiranja tekstov, iz muzejske skupnosti in ostalih sorodnih področij.

Drug možen pogled na standard Dublin Core je kot jezik, ki je namenjen postavljanju razreda izjav o virih. V tem jeziku obstajata dve skupini pojmov – elementi (samostalniki) in kvalifikatorji (pridevniki), ki jih lahko uredimo v preprost vzorec stavkov. Dublin Core lahko vidimo kot jezik, ki je zlahka naučljiv, ne zadostuje pa za izražanje bolj kompleksnih odnosov in konceptov.

Pri razvoju standarda so bili upoštevani nekateri osnovni cilji :

- preprostost izdelave in urejanja metapodatkov : Za to da bi lahko tudi nepoznavalci enostavno in poceni kreirali preproste opisne zapise informacijskih virov, je potrebno ohraniti kar se le da majhen in preprost nabor opisnih polj standarda. Istočasno je potrebno s temi polji zagotoviti učinkovito odkrivanje virov v omrežju.
- splošno razumljiva semantika : Dublin Core vsebuje splošen nabor elementov, katerih semantika je univerzalno razumljiva in podprta. S tem preprečuje potencialne razlike v terminologiji in opisnih pristopih v različnih virih znanja. Konvergenca opisov s splošnim naborom elementov poveča dosegljivost virov tako znotraj ene discipline kot tudi širše.
- mednarodni aspekt : Originalni nabor elementov Dublin Cora je bil razvit v angleščini, v zadnjem času pa se razvijajo tudi lokalizirane različice v številnih drugih jezikih. S tem je zagotovljena podpora večjezični in večkulturni naravi elektronskih informacijskih virov.

- razširljivost : Čeprav je eden glavnih ciljev standarda preprostost, pa je težnja k bolj natančnemu iskanju informacij prepričala ustvarjalce standarda, da so pričeli razmišljati tudi o mehanizmih za razširitev nabora Dublin Core elementov. Pričakuje se, da bodo tudi druge skupnosti kreirale dodatne metapodatkovne nabore, ki bi se jih potem lahko združilo z DC naborom in s tem doseglo interoperabilnost.

Opis virov s standardom Dublin Core je v praksi lahko izveden na različne načine. Ena od možnosti je neposreden generičen zapis, kot bo prikazan v opisu elementov in kvalifikatorjev v nadaljevanju, možna pa je tudi implementacija v različnih jezikih : HTML (Hypertext Markup Language), XML (eXtensible Markup Language), RDF (Resource Description Framework) ali tudi v pravem ontološkem jeziku OWL (Web Ontology Language).

1.3.2.2 Metapodatkovni format TV Anytime

TV Anytime Forum [22] je združenje organizacij, ki si želijo razviti specifikacije, ki bodo omogočale avdio vizualne in ostale storitve na osnovi digitalnih medijev. Dandanes je mogoče zelo poceni dobiti digitalne medije velikih kapacitet v najrazličnejših oblikah (diski, spominske kartice ...). To omogoča velik napredek na področju televizije, predvsem digitalne. Eden izmed bistvenih aspektov televizije, ki je najbolj pomemben za gledalce, je spored oz. program. S pojavom digitalne televizije so sporedi od preproste tekstualne oblike (teletekst) prešli v bolj kompleksno elektronsko obliko (t.i. electronic programme guides – EPG), ki je že temeljila na metapodatkih. Naslednji korak v razvoju pa, kot rečeno, pomenijo velike kapacitete poceni digitalnih medijev. Poleg vedno večjega števila televizijskih kanalov in interaktivnih storitev bodo uporabniki lahko dostopali tudi do lokalno in oddaljeno shranjenih arhivov programov. Za podporo čim bolj enostavnemu pregledovanju televizijskih kanalov in katalogov shranjenih posnetkov na digitalnih medijih je potreben ustrezen meta podatkovni zapis, ki ga bodo izkoriščale različne aplikacije, iskalni mehanizmi, navigacijska podpora in v končni fazi tudi personaliziran vodnik po sporedu oz. TV portal.

TV Anytime Forum je bil ustanovljen leta 1999 s sledečimi cilji : definirati specifikacije, ki bodo aplikacijam omogočale izkoriščati digitalne medije v uporabniških elektronskih platformah, biti neodvisen od načina dostave vsebine do uporabnika (TV, internet ...), razviti specifikacije, ki bodo združljive s sistemi ponudnikov in proizvajalcev vsebine in zagotoviti potrebne varnostne mehanizme za zaščito vseh vpletenih entitet.

V okviru TV Anytime Foruma deluje več delovnih skupin, ki se ukvarjajo z različnimi vidiki shranjevanja in dostopanja do avdio-vizualnih vsebin. Delovna skupina »Business Models« se ukvarja s preučevanjem potrebnih lastnosti in funkcionalnosti, ki bi zadovoljile potrebe različnih entitet v tej industriji – uporabnike, proizvajalce vsebin in storitev, oglaševalce in proizvajalce opreme. Skupina »Rights Managements and Protection« razvija standarde, ki bodo zagotavljali varno in fleksibilno izražanje in vsiljevanje pogojev uporabe vsebin, prenešenih do uporabnikov s strani nosilca pravic. Skupina »System, Transport Interfaces and Content Referencing« je odgovorna za skupno arhitekturo in konsistentnost TV Anytime sistema. Med drugim se ukvarja z definiranjem zahtev za spodnje transportne nivoje (slednje TV Anytime ne predpisuje), ki so potrebni za ustrezno delovanje TV A sistema, v zvezi z naslavljanjem vsebine pa velja omeniti poseben identifikator CRID (ang. content reference identifier), ki enolično določa specifično vsebino ali del le-te. Za nas najbolj zanimiva skupina »Metadata« pa je odgovorna za razvoj metapodatkovne specifikacije [24].

TV Anytime določa metapodatkovni format za meta podatke, ki se lahko izmenjujejo med različnimi entitetami, npr. med ponudnikom storitev in uporabnikom, med več uporabniki ali med uporabnikom in nekim drugim proizvajalcem metapodatkov. V okviru standarda TV Anytime so meta podatki razumljeni kot opisni podatki o avdio-vizualni vsebini, na primer naslov ali povzetek. Takšne meta podatke imenujejo atraktorji, saj uporabnike privabljajo k določeni vsebini. Omogočajo iskanje, navigacijo in urejanje vsebin iz različnih virov, med drugim interaktivne TV, interneta in lokalno shranjene vsebine. Poleg atraktorjev TV Anytime določa tudi meta podatke, ki vsebujejo informacije o uporabniških preferencah in zgodovini ogledov avdio-vizualnih vsebin. Tako definira standardne načine za opis uporabniških profilov, ki vključujejo iskalne preference za zagotavljanje avtomatskega filtriranja vsebin, ki ga v imenu uporabnikov izvajajo agenti na nivoju aplikacij. Pomemben vidik meta podatkov je tudi povezovanje teh podatkov s samo vsebino, saj je to tisti bistven del, potreben za učinkovito iskanje zanimivih vsebin. Poglejmo si sedaj nekoliko bolj podrobno v tem odstavku opisane aspekte metapodatkov, kot jih definira TV Anytime.

Eden izmed najbolj pomembnih elementov meta podatkovnega dela TV Anytime-a je CRID (Content Reference IDentifier) [25]. Gre za identifikator, ki kaže na določeno vsebino, del vsebine, lahko pa tudi na enega ali več drugih CRID-ov. Deluje tudi kot povezava med različnimi metapodatkovnimi opisi, ki se nanašajo na vsebino. CRID tako za neko enoto vsebine povezuje različne metapodatke. Pri tem TV Anytime loči 4 skupine metapodatkov :

- metapodatki za opis vsebine : Splošne informacije o določeni vsebini, ki so neodvisne od načina objave oz. predvajanja. Ti podatki opisujejo različne koncepte : najbolj preproste vsebine, vsebine, ki imajo več verzij (npr. cenzurirane verzije zaradi jezika, nasilja ...), vsebine, ki so bile zaradi narave predvajanja razdeljene v več delov (npr. triurni film razdeljen v 2 dela predvajan v 2 različnih dneh), vsebine, ki so zbirka sekvenc iz drugih vsebin, serija vsebin, ki jih lahko razvrstimo (npr. epizode), lahko pa so tudi nerazvrščene, zbirka serij ali posameznih vsebin, ki imajo isti koncept ter objava vsebine, ki ima lahko posebne atribute, ki so odvisni od načina objave. Glavne entitete modela za opis vsebine so : vsebina (koherentna enota vsebina, ki je lahko sestavljena iz več drugih vsebin), skupina vsebin (predstavlja način grupiranja vsebin – serija, show ...), lokacija vsebine (informacija o lokaciji enega primerka vsebine).
- metapodatki za opis primerkov vsebine : Lokacija vsebine, načini uporabe, parametri za dostavo – format vsebine ... Ti podatki se uporabljajo, ko obstajajo bistvene razlike med primerki istega tipa vsebine, to so primerki, ki imajo isti CRID. Ti metapodatki so povezani s konkretnim primerkom vsebine, ki je odvisen od določenega dogodka. Tudi tu je nekaj osnovnih entitet : lokacija vsebine (generična lokacija vsebine ne glede na naravo medija), storitev (specifičen tok predvajane vsebine, ki poteka po nekem fizičnem kanalu), dogodek »Schedule« (poseben tip vsebine, ki je primeren za opisovanje lokacij predvajanih vsebin).
- metapodatki o uporabniku : Delijo se na uporabniške preference, ki se uporabljajo za iskanje, filtriranje, izbiro in uporabo vsebin. V teh zapisih uporabnik določi npr. kakšen žanr vsebine mu je všeč, jezik, državo, naslov vsebine, najljubšega igralca itd. Druga skupina metapodatkov pa so podatki, ki opisujejo uporabnikovo zgodovino uporabe vsebin. Podatki o tem se zbirajo preko daljšega časovnega obdobja.
- segmentacijski metapodatki : Ti podatki se nanašajo na sposobnost definirati, dostopati in obdelovati segmente oz. časovne intervale določene vsebine znotraj avdio vizualnega toka. Vsebujejo npr. povzetek vsebine z bistvenimi poudarki, niz zaznamkov, ki kažejo na specifične dele toka itd.

Od naštetih skupin metapodatkov se osredotočimo le na prvo skupino, ki je namenjena opisu vsebine nekega programa. Od številnih možnih opisnih elementov TV Anytime predpisuje obvezno uporabo polja za opis naslova, priporočljivo pa je uporabljati tudi polja za opis žanra vsebine, povzetka vsebine (sinopsisa), jezika (govorjen jezik v filmu, jezik podnapisov, jezik avdio zapisa vsebine) in seznam najbolj pomembnih oseb, povezanih z vsebino (režiser, scenarist, ponudnik, glavni igralec ...). TV Anytime uporablja 4 nivojski sistem klasifikacije vsebin glede na vsebino. Glavna kategorije so : Information, Drama, Entertainment, Music, Enrichment, Movies, Animations/ special effects, Hobby, Sports events, Pure information, Information / tabloid, Documentary, Education in Children.

Metapodatkovna specifikacija, ki jo definira TV Anytime, je izražena v obliki metapodatkovne sheme. To je formalna definicija strukture in tipa meta podatkov. TV Anytime za opis strukture metapodatkov uporablja MPEG 7 Description Definition Language [21], za format predstavitve metapodatkov pa uporablja XML (eXtensible Markup Language) [26], ki zaradi splošne rabe zagotavlja interoperabilnost, poleg tega pa je tudi razširljiv. Velja opozoriti, da XML ni edini možen način predstavitve, je pa uradno definiran v TV Anytime specifikacijah.

Za boljšo predstavo, kako izgleda opis neke vsebine z TV Anytime metapodatkovnih zapisom, si poglejmo manjši zgled, kjer je prikazana shema za opis žanra vsebine :

```
<complexType name="GenreType">
	<complexContent>
		<extension base="tva:ControlledTermType">
			<attribute name="type" use="optional" default="main">
				<simpleType>
					<restriction base="string">
						<enumeration value="main"/>
						<enumeration value="secondary"/>
						<enumeration value="other"/>
					</restriction>
				</simpleType>
			</attribute>
		</extension>
	</complexContent>
<complexType>
```

1.3.2.3 MPEG-7

MPEG-7 je standard za opis in iskanje avdio in vizualnih vsebin, ki ga je razvila MPEG (Moving Picture Experts Group) [21, 27] in je veljaven ISO/IEC standard. Njegov namen je zagotavljanje dodatne funkcionalnosti prejšnjim MPEG standardom, in sicer informacij o vsebinah, ne pa samim vsebinam (vrsti kodiranja ipd.). S standardizacijo opisov multimedijskih vsebin je omogočeno hitro in učinkovito iskanje zanimivega avdio-vizualnega materiala.

Standard MPEG-7 standardizira:

- Nabor opisnih shem (Description Schemes) in opisnih elementov (Descriptors).
- Jezik, ki definira te sheme (Description Definition Language).
- Sheme za kodiranje opisov.

Razvijalci standarda so si pri določanju standarda postavili več ciljev, ki naj bi jih standard dosegel:

- Omogočanje hitrega in učinkovitega iskanja, filtriranja in identifikacije vsebin.
- Opis glavnih lastnosti vsebin (nizkonivojske karakteristike, struktura, modeli, zbirke …).
- Registriranje širokega spektra aplikacij.
- MPEG-7 obravnava avdio-vizualne informacije: avdio, zvok, video, slike, grafi in 3D modeli.
- Sporočanje, kako so objekti urejeni na sceni.
- Neodvisnost med opisi in vsebinami.

Zadnji cilj določa, da morata biti po MPEG-7 arhitekturi opis vsebine in vsebina ločena. Po drugi strani pa mora obstajati povezava med njima, zato je opis običajno multipleksiran s samo vsebino.

Standard MPEG 7 je sestavljen iz različnih delov:

1. MPEG-7 Systems: orodja, ki so potrebna za pripravo MPEG-7 opisov za učinkovit prenos in shranjevanje. Ukvarjajo se tudi z zaščito intelektualnih pravic.
2. MPEG-7 Description Definition Language: jezik, ki se uporablja za definicijo sintakse MPEG-7 orodij za opis (Description Tools) in za definicijo novih opisnih shem (Description Schemes). Temelji na jeziku XML.
3. MPEG-7 Visual: opisna orodja (Description Tools), namenjena vizualnim opisom.
4. MPEG-7 Audio: opisna orodja (Description Tools), namenjena audio opisom.
5. MPEG-7 Multimedia Description Schemes: opisna orodja (Description Tools), namenjena splošnim značilkam in multimedijskim opisom.
6. MPEG-7 Reference Software – programska implementacija ustreznih delov standarda MPEG-7.

7. MPEG-7 Conformance Testing – navodila in postopki za testiranje skladnosti MPEG-7 implementacij.
8. MPEG-7 Extraction and use of descriptions: informativno gradivo (v obliki tehničnega poročila) o izvlečkih in uporabi nekaterih opisnih orodij.
9. MPEG-7 Profiles and levels: navodila in standarni profili.
10. MPEG-7 Schema definiton: določa shemo z uporabo z jezika za definicijo opisa (Description Definition Language).

Standard MPEG-7 se uporablja na številnih področjih in aplikacijah. Naštejmo nekaj primerov:

- Digitalne knjižnice: slikovni/video katalogi, glasbeni slovar.
- Storitve multimedijskih kazal: npr. rumene strani.
- Izbira prenosnih medijev: radiokanal, TV kanal.
- Urejanje multimedije: personalizirane elektronske novice, lastništvo vsebin.
- Varnost: nadzor prometa, proizvodnje verige.
- Kultura: umetnostne galerije, muzeji …
- Izobraževalne aplikacije.

Za konec te kratke predstavitve MPEG-7 standarda si oglejmo še preprost primer opisa neke slike:

```
<Creation>
   <Title type="original">
      <TitleText xml:lang="si">
         Narava
      </TitleText>
      <TitleImage>
         <MediaURL>file://images//narava.jpg></MediaURL>
      </TitleImage>
   </Title>
   <Creator>
      <role>photographer</role>
      <GivenName>Ana</GivenName>
      <FamilyName>Blatnik</FamilyName>
   </Creator>
   <CreationDate>
      <D>16</D>
      <M>2</M>
      <Y>2009</Y>
   </CreationDate>
   <CreationLocation>
      <PlaceName xml:lang="si">Ljubljana</PlaceName>
      <Country>SI</Country>
   </CreationLocation>
</Creation>
```

1.4 Uporabniški model

Ker je med uporabniku prilagojenem iskanju vsebin potrebno analizirati veliko količino podatkov, ni smotrno procesa v celoti ponavljati vsakič, ko se uporabnik prijavi v sistem. Zato se je pojavila težnja po tem, da bi se vmesni rezultati sistema za iskanje vsebin shranjevali v namenski objekt. Ker se proces iskanja uporabniku primernih vsebin pogosto obravnava tudi kot uporabniško modeliranje (ang. User modeling), se je objekt poimenoval uporabniški model oziroma uporabniški profil.

V uporabniški profil se shranjujejo splošni (ime, priimek, spol, starost, elektronski naslov, itd.) ter tehnični podatki, ki jih sistem potrebuje za pravilno iskanje vsebin. Podatki se razlikujejo od sistema do sistema oz. od pristopa do pristopa. Zato se lahko uporabniški modeli med seboj zelo razlikujejo in v veliki večini primerov tudi niso kompatibilni. Če dva sistema delujeta v različnih okoljih (na primer prvi za predlaganje spletnih dokumentov, drugi pa za iskanje primernih filmov na televizijskem programu), njuna uporabniška modela nista medsebojno izmenljiva. Če uporabnik tako na primer prvi sistem uporablja že dalj časa in ima v njem že izdelan natančen model, mu ta ne koristi, če želi začeti uporabljati še drugi sistem. Namesto tega se mora v drug sistem prijaviti kot popolnoma nov uporabnik. Na področju združljivosti oziroma prenosljivosti uporabniških modelov poteka veliko raziskav, vendar trenutno še ni na voljo uporabna rešitev tega problema.

1.5 Struktura dela

Drugo poglavje z naslovom 'Uporabniku prilagojeno iskanje vsebin' vsebuje predstavitev področja personalizacije, problemov, pristopov in metod, ki jih srečujemo v okviru tega področja.

V tretjem poglavju z naslovom 'Testno okolje' opisujemo testno okolje, v katerem smo razvijali nove pristope, tehnike, s pomočjo katerih smo razvite pristope ovrednotili, ter podatke, s pomočjo katerih smo lahko pristope testirali.

V četrtem poglavju z naslovom 'Razvoj sistema za skupinsko filtriranje' opisujemo prvi sistem za uporabniku prilagojeno iskanje vsebin, razvit v okviru tega doktorskega dela. Predstavljeni so zasnova sistema, uporabljeni pristopi ter inovacije, predvidena uporaba sistema ter možne izboljšave za nadaljnji razvoj.

V petem poglavju z naslovom 'Izbira vrednosti parametrov sistema za skupinsko iskanje primernih vsebin' predstavimo ključne parametre razvitega sistema ter postopke kako, poiskati vrednosti teh parametrov, ki zagotovijo optimalno delovanje sistema.

V šestem poglavju z naslovom 'Kombiniran vsebinsko-skupinski pristop' predstavimo kombiniran sistem za uporabniku prilagojeno iskanje vsebin, ki je bil razvit kot nadgradnja sistema iz tretjega poglavja. Opisane so inovacije, dodani pristopi ter njih implementacija. Poleg tega je opisana tudi struktura novega sistema.

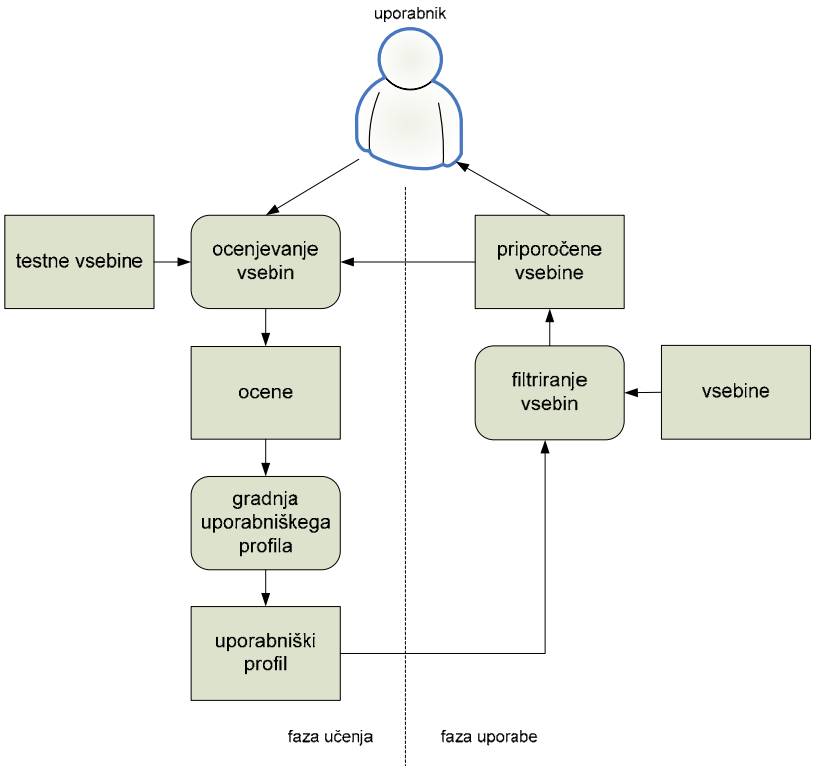
Sedmo poglavje pa predstavlja zaključek, v katerem predstavimo ugotovitve, do katerih smo prišli tekom raziskav, izvirne prispevke znnanosti ter možnosti za nadaljnji razvoj.

2. Uporabniku prilagojeno iskanje vsebin

V okviru tega poglavja bomo podrobneje opisali področje uporabniku prilagojenega iskanja vsebin, splošne pojme ter težave, na katere naletimo na tem področju. Predstavili bomo tudi problematiko, s katero se ukvarja pričujoče doktorsko delo.

2.1 Splošen pregled področja in osnovnih pojmov

Informacijska doba je uporabniku omogočila, da dostopa do vsebin kjerkoli in kadarkoli. Posledica lažjega in obsežnejšega dostopa vsebin je tudi strma rast samega števila dostopnih vsebin. Uporabnik se zato znajde v težavni situaciji, ker ima na voljo informacije o praktično vsem, kar ga zanima, hkrati pa nima na voljo načina, kako bi do teh informacij prišel hitro in enostavno. Uporabnik zato ni več sposoben sam učinkovito iskati informacij, temveč potrebuje sistem za avtomatizirano iskanje vsebin. Tak sistem preišče vse dostopne vsebine, oceni njihovo primernost glede na potrebe uporabnika in na koncu uporabniku ponudi selekcijo vsebin, za katere meni, da bodo uporabniku najbolj koristne. Za pravilno delovanje mora seveda poznati uporabnikove navade in želje.



Slika 4: potek UM (učenje, model, recomm)

Zaradi kompleksnosti problematike uporabniku prilagojenega iskanja multimedijskih vsebin smo se odločili, da se pri implementaciji sistema omejimo na področje personaliziranega iskanja multimedijskih vsebin s pristopom skupinskega filtriranja.

2.1.1 Uporabniški model

Glavni namen uporabniškega modela je shranjevanje podatkov, ki jih sistem za uporabniku prilagojeno iskanje vsebin potrebuje za pravilno delovanje. Podatki obsegajo osnovne informacije o uporabniku (spol, starost, ime in priimek itd.), podatke o zgodovini uporabnikove interakcije z vsebinami (ang. usage history), uporabnikovem mnenju o vsebinah (ocene, ankete) in specifične podatke, ki so odvisni od uporabljene metode. Obstaja več različnih pristopov uporabniku prilagojenega iskanje vsebin, zato uporabniški modeli niso enaki za vse sisteme. Če je sistem osnovan na iskanju podobnosti med posameznimi vsebinami, bo uporabniški model vseboval podatke o tem, kateri tipi vsebin so uporabniku všeč in kateri ne. V sistemu, osnovanem na iskanju uporabnikov s podobnim okusom, pa bo uporabniški model lahko vseboval samo podatke o tem, kateri uporabniki imajo podoben okus. Uporabniški modeli zato nastopajo v različnih oblikah in podatkovnih strukturah. To je tudi vir težav, s katerimi se srečujejo sistemi za uporabniku prilagojeno iskanje vsebin; prenosljivost uporabniških modelov med sistemi je slaba, oziroma v večini primerov celo nemogoča.

Uporabniški model je lahko samostojna podatkovna struktura, ločena od samega algoritma [14, 15, 16, 27], lahko je del samega izvajanega algoritma [13, 14] ali pa je preprosto predstavljen kot samo množica preteklih dejanj uporabnika v obliki dostopanih in eksplicitno ocenjenih vsebin.

Ne glede na strukturo je glavni namen uporabniškega modela, da hrani zgodovino uporabnikove interakcije z vsebinami in uporabnikove preference, odkrite na podlagi teh interakcij.

Sistem najprej razvije uporabniški model v fazi učenja, kjer se uporabniški model oblikuje na podlagi že zabeleženih uporabnikovih interakcij (t.i. učna množica). S pomočjo teh podatkov sistem razvije uporabniški model, v katerega zapiše vse pomembne parametre, s pomočjo katerih bo kasneje lahko iskal primerne vsebine. Ko ima sistem na voljo začetni profil uporabnika, lahko preide v fazo uporabe, kjer uporabnikov model uporabi za iskanje primernih vsebin. Na podlagi povratnih informacij, pridobljenih s strani uporabnikov, sistem profil posodablja in tako poskuša uporabniku vedno ponuditi najprimernejše vsebine.

2.1.2 Učenje uporabniškega modela

Sistem ne more iskati vsebin, primernih za uporabnika, dokler ne ve, katere vsebine so uporabniku sploh zanimive. To lahko ugotovi samo na osnovi poznavanja uporabnikovega modela, ki ga pridobi z analizo uporabnikove zgodovine uporabe sistema.

Prva faza vsakega sistema za uporabniku prilagojeno iskanje vsebin je tako gradnja oziroma učenje uporabniškega modela. Ne glede na uporabljeno metodo gre v tej fazi za analizo obstoječih podatkov o uporabnikih, vsebinah ter interakcijah med uporabniki in vsebinami. Faza učenja lahko pravilno poteka le na osnovi podatkov, ki so bili že predhodno zbrani. Celotna infrastruktura mora omogočati slednje uporabniških dejanj ter hranjenje podatkov.

2.1.3 Izbira primernih vsebin za uporabnika

Ko ima sistem na voljo uporabniški model aktivnega uporabnika, lahko začne z iskanjem primernih vsebin. Proces iskanja primernih vsebin je zelo odvisen od uporabljenih pristopov. Najbolj pogosto uporabljana pristopa sta vsebinski in skupinski [28].

Pri vsebinskem pristopu sistem primerne vsebine išče tako, da iz uporabniškega modela razbere, kateri tipi (žanri, igralci, režiserji, itd.) uporabnika najbolj zanimajo, in nato na podlagi opisa za vsako vsebino oceni, v kolikšni meri bi potencialno zanimala uporabnika.

Pri skupinskem pristopu sistem iz uporabniškega modela razbere, kateri uporabniki imajo podoben okus kot aktivni uporabnik, in nato preveri, katere vsebine so jim bile všeč, nato pa izbor teh vsebin predlaga aktivnemu uporabniku.

Ne glede na uporabljeni pristop mora sistem v tej fazi vedno analizirati vsebine, ki jih aktivni uporabnik še ni videl in za vsako izmed njih ugotoviti, ali je primerna za uporabnika ali ne. Končni rezultat te faze je lista vsebin in izračunane primernosti za vsako izmed njih. Seveda pa sistem ne ponudi celotne liste uporabniku, temveč na koncu vedno ponudi samo ožji izbor, ki lahko obsega od pet do deset vsebin oziroma v nekaterih primerih celo eno samo vsebino.

Vsebine se uporabniku lahko ponudijo v obliki spiska ali pa včasih (predvsem v primeru multimedijskih vsebin) kot uporabniku prilagojen elektronski program (ang. *Electronic programme guide*), ki se vedno bolj uporablja v sklopu digitalne in spletne televizije.

Sistem nato zabeleži, če je uporabnik katero izmed ponujenih vsebin izbral in kako je bil z njo zadovoljen.

2.1.4 Zbiranje povratnih informacij

Sistem uporabniku prilagojenega iskanja vsebin torej deluje tako, da analizira dosedanje uporabnikove interakcije z vsebinami in na podlagi te analize ugotovi, katere vsebine so primerne ter katere ne. Iz tega je razvidna tudi zahteva, da mora sistem biti sposoben slediti uporabnikovim interakcijam z vsebinami.

Na tem področju obstajata dva pristopa – eksplicitni in implicitni:

- Eksplicitni pristop je zasnovan na tem, da sistem uporabnika neposredno prosi za povratno informacijo, ponavadi v obliki ocene (na primer z lestvico od ena do deset) ali vprašalnika.
- Implicitni pristop je bolj prijazen do uporabnika, saj lahko deluje povsem neopazno, vendar pa v zameno za to ne more delovati z veliko stopnjo zanesljivosti. Implicitni pristop temelji na pasivnem opazovanju uporabnika ter sledenju njegovim dejanjem. Kot primer lahko navedemo ogled video vsebin, kjer sledimo, ali je uporabnik odprl datoteko, je uporabil premor, je pogledal vsebino do konca. Na podlagi teh podatkov mora sistem nato interpretirati, ali je uporabnik vsebino pozitivno sprejel ali ne. Če je vsebino odprl ter jo pogledal do konca, bi lahko sklepali, da mu je bila všeč, če pa jo je takoj ustavil, pa lahko sklepamo, da je bil njegov odziv negativen.

Eksplicitni pristop nam omogoča natančnejše delo, saj imamo točne podatke o uporabnikovih preferencah. Vendar pa v zameno za to uporabnika pogosto motimo oz. ga prosimo za povratno informacijo. Implicitni pristop pa je bolj prijazen do uporabnika, saj lahko deluje tako, da se uporabnik sploh ne zaveda, da obstaja. V zameno zato pa potrebujemo več truda, da začne sistem pravilno in zanesljivo delovati, saj vse gradimo na predvidevanju, da znamo uporabniška dejanja pravilno interpretirati.

2.2 Terminologija in matematične oznake

Na področju uporabniku prilagojenega iskanja vsebin vsaka raziskovalna skupina uporablja svoje oznake, ker še ne obstaja noben standard za zapis enačb na področju personalizacije. V doktorski disertaciji bomo uporabljali za zapis algoritmov ter enačb oznake in pojme, predstavljene v članku [29]. Začetne oznake iz članka smo nadgradili ter razširili v upanju, da bomo tako sčasoma prispevali k razvoju novega standarda.

Pri delu z uporabniku prilagojenimi sistemi imamo opravka z večjim številom uporabnikov ter vsebin, zato je najprej potrebno definirati, kako bomo obravnavali skupine ter posameznike.

Privzamemo, da nam vsi uporabniki v obstoječem podatkovnem sistemu tvorijo množico uporabnikov, ki vsebuje vse njihove profile. To množico označimo z U, posameznega uporabnika pa označimo kot $u \in U$. Množico vseh vsebin, s katerimi sistem deluje, označimo s H, posamezno vsebino pa s $h \in H$.

Ker nas pogosto zanima, kako je specifičen uporabnik ocenil izbrano vsebino, uporabimo v takem primeru zapis $e(u_i, h_k)$, kjer nam u_i predstavlja uporabnika *i*, ki je ocenil izbrano vsebino h_k.

Da priporočilni sistem lahko deluje, mora vsakemu uporabniku zgraditi uporabniški model, ki vsebuje podatke, potrebne za iskanje primernih vsebin. Uporabniški model uporabnika u označimo z $um(u)$. Izgradnjo modela označimo s preslikavo $um: U \rightarrow UM$.

Sistem nam za vsako potencialno zanimivo vsebino vrne oceno $\hat{e}(u_i, h_k)$, ki predstavlja napovedano oceno, ki opisuje, kako naj bi se uporabnik na predlagano vsebino odzval. Dejansko vrednost ocene, ki jo uporabnik dodeli vsebini pa označimo z $e(u_i, h_k)$.

Za vsako vsebino imamo na voljo opis z metapodatki. Metapodatke vsebine h označimo z $md(h)$. Vsaka vsebina je lahko opisana z več atributi A. S pomočjo atributov metapodatke razdelimo v tematske podskupine (na primer žanr, direktorji, kategorija itd.). Primer opisovanja vsebine z atributi je razviden na tabeli 1, kjer je opisana vsebina iz slike 5.

Atribut	Vrednost atributa
Žanr	Drama
Direktor	Christopher Hampton
Igralci	Bob Hoskins, Patricia Arquette, Gerard Depardieu
Država snemanja	Velika Britanija
Leto snemanja	1996

Tabela 1: Opis vsebine z atributi

Osredotočili smo se predvsem na podatke o žanru. Vsaka vsebina je lahko opisana z več različnimi žanri. Žanr predstavimo kot vrednost atributa $a_i \in A$, kjer nam vsak a_i predstavlja en obstoječ žanr (npr a_1 predstavlja fantasy, a_2 dramo, a_3 komedijo itd.).

Pri delu z našim kombiniranim sistemom smo uvedli pojem uporabnikove žanrske preference. Žanrska preferenca nam predstavlja porabikovo mnenje o posameznem obstoječem žanru (torej da ima na primer rad fantastiko in sovraži dramo). Označimo jo z $ap(u_i, a_j)$, kjer u_i predstavlja uporabnikov profil, a_j pa žanr. Vrednost te mere predtsavlja uporabnikovo mnenje o žanru (več o tem v poglavju 6.4) .

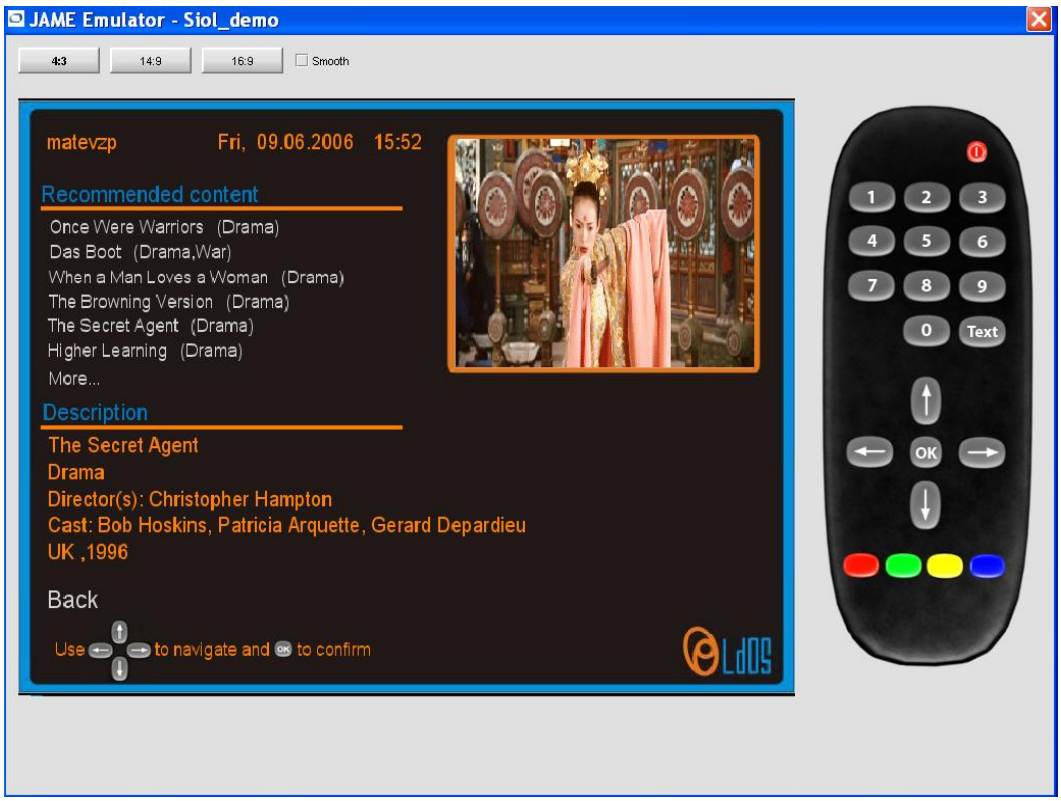
2.3 Področja uporabniku prilagojenih storitev

Prve uporabniku prilagojene storitve so razvili v okviru spletnih storitev kot pametni iskalniki spletnih dokumentov [2, 3, 4, 5]. Z vedno večjim prodorom spleta med uporabnike in razvojem novih tehnologij pa so se uporabniku prilagojene storitve razširile tudi na veliko drugih področij, kot na primer spletno trgovanje, personalizirano oglaševanje, digitalna televizija ter spletno hranjenje vsebin.

2.3.1 Interaktivna digitalna TV

S pojavom digitalne televizije, ki za delovanje potrebuje tudi spletno povezavo s centralo, se je pojavila želja po uvedbi prilagojenih storitev tudi na tem področju. Uporabnik do digitalne televizije dostopa preko vmesne naprave, priključene med centralo ter uporabnikovo televizijo. Te vmesne naprave se imenujejo Set-top-box (STB) in so v resnici preko spleta povezane s centralo operaterja, kjer hranijo multimedijske vsebine. Brez uporabniku prilagojenih storitev poteka interakcija s sistemom tako, da ponudnik (npr. kabelski operater) iz centrale oddaja programe v digitalni tehniki, uporabnik pa lahko nato samo preklaplja med obstoječimi programi. Toda ker se ti programi oddajajo preko spleta in ker splet omogoča naprednejše storitve, so se pojavile želje po uvedbi uporabniku prilagojenih storitvah.

Prva in bolj osnovna storitev na tem področju je uvedba personaliziranega programskega vodiča (ang. EPG – electronic personal guide), ki uporabnika opozori na program in čas, ob katerem se bo na njem nahajala vsebina, ki bi uporabnika zanimala. Na sliki 5, je prikazan primer Siolovega programskega vodiča, s prikazom podatkov o izbrani vsebini.



Slika 5: Personaliziran programski vodič

Pri razvoju teh storitev je bilo potrebno upoštevati tudi določene tehnološke omejitve, saj STB ne omogočajo naprednih uporabniških vmesnikov in mehanizmov povratnih informacij.

Naslednji nivo uporabniku prilagojenih storitev so razvili pri ponudnikih, ki omogočajo tudi dostop do video vsebin na zahtevo uporabnika (ang. *Video on demand*). Pri teh sistemih lahko uporabnik kadarkoli zahteva katerokoli izmed vsebin, hranjenih v centrali, nakar se ta vsebina pretoči k uporabniku. Težava teh sistemov je seveda sama količina vsebin, ki je na voljo uporabniku. Zato so se pojavili priporočilni sistemi, ki uporabniku pomagajo hitreje najti primerne vsebine.

2.3.2 Spletno hranjenje multimedijskih vsebin

Z večanjem kapacitete, ki je na voljo za spletno hranjenje multimedijskih vsebin (ang. web repository), narašča tudi količina spletnih strani, ki nudijo storitve shranjevanja ter predvajanja poljubnih uporabniških vsebin (Youtube [30], shelfari [31], facebook [32]). Hitro narašča tudi število uporabnikov, ki te storitve uporabljajo in posledično tudi količina vsebin, ki so na voljo na takih straneh. Zato se uporabniki ponovno srečujejo s težavo iskanja vsebin, ki bi jih zanimale, saj so lahko količine podatkov res velike (na primer več kot 3 bilijone video posnetkov na Youtube).

2.3.3 Reklamna sporočila

Obetavno področje za uporabniku prilagojene storitve so reklamna sporočila. Na spletnih straneh (in tudi na televizijskih programih) je uporabnik pogosto zasut z veliko reklamnih sporočil. Uporabniku v večini primerov to ni všeč iz dveh razlogov:

- Količina reklamnih sporočil – če je na spletni strani preveč reklamnih sporočil, lahko postane ostala vsebina nepregledna.
- Tip reklamnih sporočil – vsako reklamno sporočilo je narejeno za določeno ciljno skupino uporabnikov. Če uporabnik tej skupini ne pripada, mu je lahko reklamno sporočilo moteče oz. ga lahko odvrne od ostale vsebine.

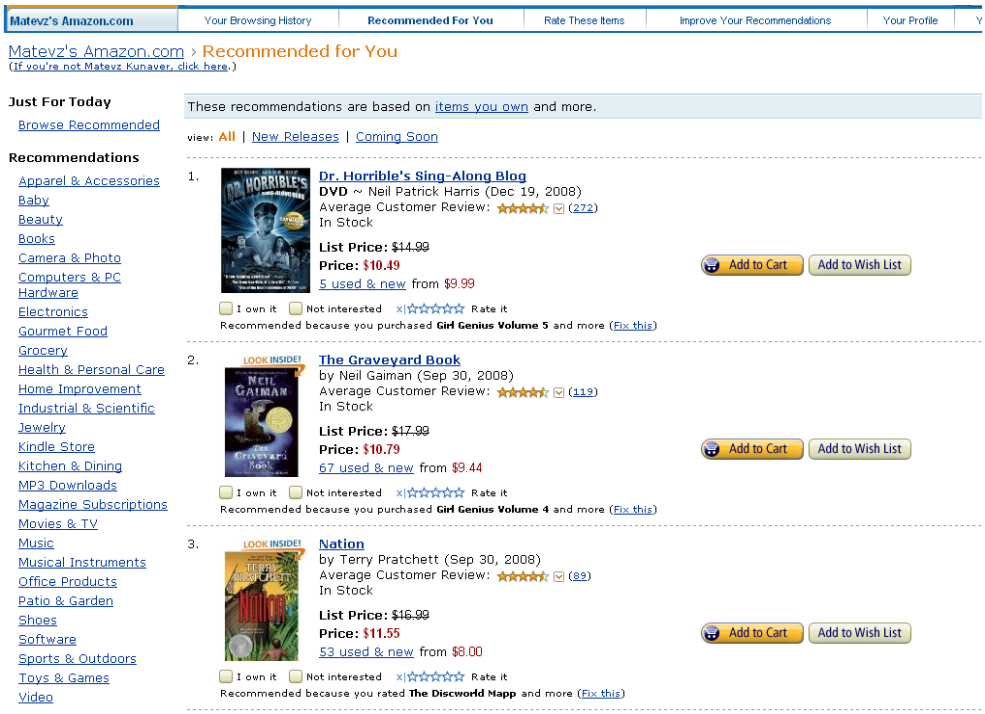
Z uvedbo uporabniku prilagojenih reklamnih sporočil bi lahko odpravili obe težavi. Ker bi uporabnik videl samo reklame, ki bi ga zanimale, bi bil potencialni uspeh reklame boljši, saj bi bila večja verjetnost, da bi se uporabnik pozitivno odzval na reklamo in kupil oglaševani artikel oziroma storitev. Poleg tega uporabnika ne bi bilo potrebno zasipati z veliko količino reklamnih sporočil hkrati, temveč bi se mu lahko zanj zanimiva sporočila predvajalo v zaželenih intervalih in tako za uporabnika sploh ne bi bila moteča.

2.3.4 Trgovine

Ko je pred uporabnika v trgovini postavljena velika množica artiklov, vedno nastane težava katere artikle izbrati. Če uporabnik točno, ve kaj hoče, to ni težava, če pa želi poskusiti kaj novega, pa ga lahko količina ponudbe prestraši.

Personalizirane storitve mu zato lahko pri tem zelo pomagajo. Te storitve so trenutno predvsem na voljo pri spletnih storitvah, počasi pa se širijo tudi v 'prave' trgovine. Spletne trgovine imajo na voljo podatkovno bazo artiklov, ki jih je uporabnik do sedaj pri njih že kupil in lahko na podlagi te zgodovine nakupov ugotovijo, kateri artikli bi bili v prihodnosti zanimivi zanj. Dober primer take storitve je prikazan na sliki 6, kjer spletna trgovina Amazon

[11], ponuja artikle na podlagi preteklih nakupov ter nakupov uporabnikov s podobnim okusom ("kupci, ki so kupili isti artikel, so kupili tudi….").



Slika 6: Spletni vmesnik trgovine amazon.com s primerom predlaganih vsebin

2.4 Postopki in metode za ocenjevanje primernosti vsebin ter njihovo razvrščanje

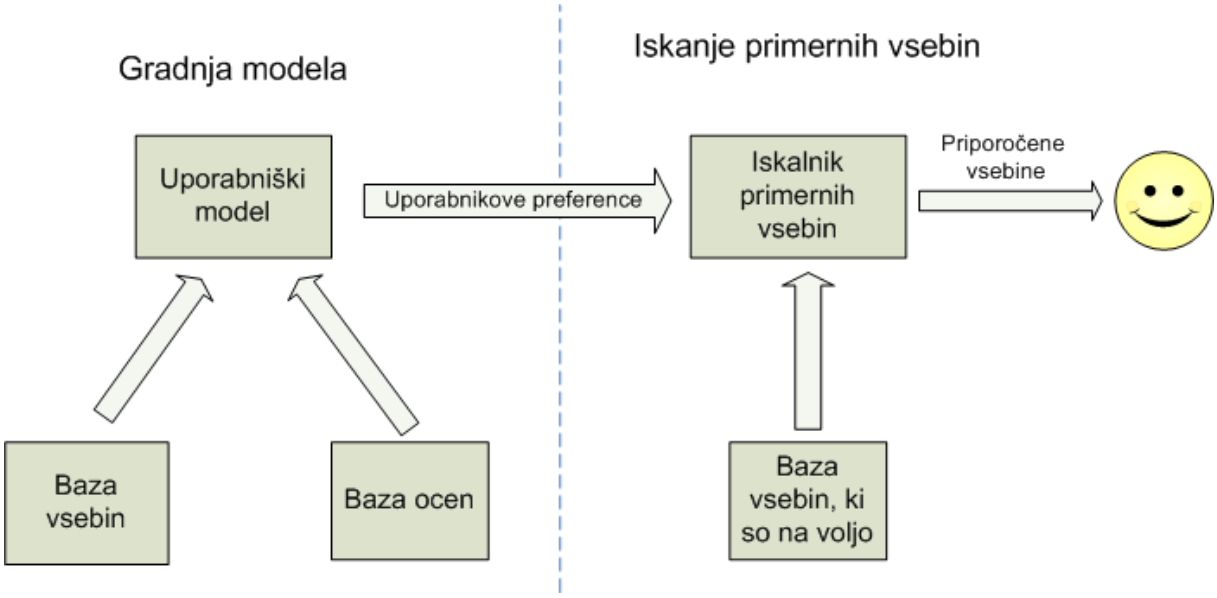
Kot smo že omenili, obstaja veliko različnih sistemov uporabniku prilagojenega iskanja vsebin, ki se med seboj razlikujejo glede na uporabljene pristope. Poudariti je potrebno, da sisteme ne delimo samo na osnovi uporabljenega pristopa, temveč jih še prej delimo na podlagi osnovnega principa filtriranja vsebin. Znotraj posameznega principa filtriranja vsebin je razvitih mnogo različnih pristopov, zato najprej predstavimo osnovne principe. Najbolj razširjena ločimo glede na to, na kaj se sistem osredotoči pri iskanju primernih vsebin. Če se sistem osredotoča na uporabnike, potem govorimo o skupinskem filtriranju, če pa se namesto tega osredotoča na posamezne vsebine, pa govorimo o vsebinskem filtriranju. Če je sistem zasnovan kot kombinacija več pristopov (ne glede na to iz katere skupine), govorimo o kombiniranih sistemih. Ti trije principi so najbolj pogosto omenjani v literaturi in jih bomo zato v sledečih poglavjih bolj podrobno predstavili. Seveda pa obstaja še veliko drugih skupin, ki jih bomo na kratko predstavili v podpoglavju 2.4.4.

2.4.1 Vsebinsko filtriranje

Vsebinsko ali individualno filtriranje (ang. *content-based recommender*) je pristop, pri katerem se primerne vsebine išče izrecno na podlagi uporabnikovega 'okusa' oziroma njegovih preferenc. Preference ugotovimo na podlagi ugotavljanja podobnosti med metapodatkovnimi $md(h)$ (ang. *meta-data*) opisi vsebin (visoko-nivojske lastnosti) ter uporabniškim modelom posameznega uporabnika $up(u)$ [15, 27, 33, 34, 35]. Tak sistem lahko deluje le na osnovi predhodno definiranih mer podobnosti med opisi vsebin in

uporabniškimi modeli. Najpogosteje pri iskanju spletnih dokumentov in uporabniških modelov uporabljamo vektorje besed, za mere podobnosti pa kosinusno ali evklidsko razdaljo med njimi [2]. Ta pristop predstavlja tudi osnovo za iskanje multimedijskih vsebin. Razširjen je tudi način, ki predstavlja uporabniške modele v obliki drevesnih struktur, v katerih je vsebovana informacija o priljubljenosti posameznih kategorij [34, 35], v katere so razvrščeni dokumenti. Mera podobnosti s posameznim dokumentom je določena kot vsota produktov med utežmi kategorij, v katere spada dokument in utežmi istih kategorij v uporabniškem modelu. Pri iskanju televizijskih in radijskih programov obstaja tudi [33] primer, kjer je uporabniški model predtsavljen kot shema, ki je primerna za primerjavo z opisi posameznih programov. Mera podobnosti je utežena vsota mer podobnosti na nivoju posameznih lastnosti (žanr, ključne besede, itd.). Uporabniški model lahko predstavimo tudi kot nabor kategorij in pripadajočih ključnih besed s pozitivno ali negativno oznako, ki predstavljajo uporabnikov interes za posamezno kategorijo [16]. Ustreznost posamezne vsebine se izračunava s pomočjo naivnega Bayesovega klasifikatorja [2].

Ne glede na izbrane metode iskanja je pri vsebinskem filtriranju bistveno to, da so vsebine izbrane in predlagane uporabniku le na podlagi primerjave z njegovim uporabniškim modelom in ne na podlagi primerjave uporabnikovega okusa z drugimi uporabniki. Sistem zato deluje tudi v razmerah z malo aktivnimi uporabniki, saj se vsak uporabnik obravnava kot neodvisna celota.



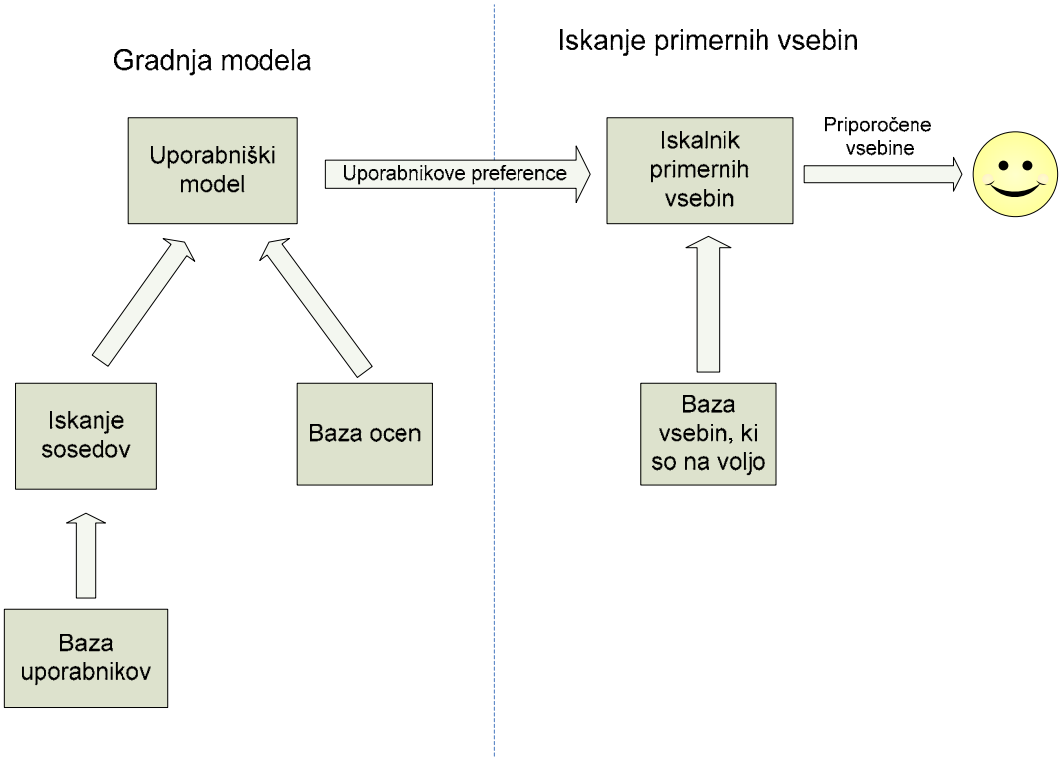
Slika 7: Vsebinsko filtriranje

Na sliki 7 je splošni potek vsebinskega filtriranja. Uporabniški model je sestavljen na podlagi analize uporabnikove zgodovine (zabeležene kot ocene vnesene v bazi ocen), ter lastnosti vsebin, ki jih je uporabnik ocenil (lastnosti so zabeležene v bazi vsebin). Ko je potrebno uporabniku predlagati primerne vsebine, se uporabnikov model naloži v iskalnik primernih vsebin, ki na podlagi podatkov vnesenih v uporabnikovem modelu iz baze vsebin, ki so na voljo (torej vsebine, do katerih uporabnik še ni dostopal).

2.4.2 Skupinsko filtriranje

Skupinsko ali socialno filtriranje (ang. collaborative recommender) deluje na principu primerjanja uporabnikov oz. njihovih modelov. Pri tem načinu skušamo uporabnike razvrstiti v eno izmed obstoječih skupin podobnih uporabnikov [35, 36, 37, 38, 39] ali pa vsakemu uporabniku poiščemo skupino N-najbližjih sosedov [33, 40]. Za uporabnika potencialno primerne vsebine so tiste, ki so jih sosedje že ocenili, uporabnik pa ne. Izmed teh vsebin nato z uporabo filtrirnih metod izberemo najbolj primerne in jih predlagamo uporabniku. Za računanje podobnosti med uporabniki se najbolj pogosto uporablja pristop, kjer je mera podobnosti med uporabniki odvisna od ocen istih vsebin [33, 35, 36, 40, 41]. Ta pristop je problematičen, ker je predvsem na začetku delovanja sistema na voljo zelo malo vsebin, ki bi jih ocenilo ali izbralo več uporabnikov. Delna rešitev tega problema je nadomeščanje manjkajočih ocen s privzetimi vrednostmi [36], iskanje podskupin uporabnikov, ki so ocenjevali iste vsebine, ali z uvrščanjem uporabnikov v stereotipne skupine glede na demografske podatke [37, 38, 39] ali na anketo, izpolnjeno ob prijavi.

V nasprotju z vsebinskim filtriranjem so pri skupinskem vsebine izbrane in predlagane uporabniku le na podlagi ocen njegovih sosedov oz. uporabnikov, ki se nahajajo v isti skupini. Zato so sistemi, osnovani na skupinskem filtriranju, lahko zelo fleksibilno zasnovani ter omogočijo zasnovo sistema, ki pokriva več različnih tipov vsebin. Pri vsebinskem pristopu to ni možno, saj sistem deluje na podlagi primerjave uporabniškega modela z metapodatki vsebin in zato ne more uporabiti istega profila za vsebine, opisane z drugačnim metapodatkovnim standardom. Sistem za skupinsko filtriranje vsebin pa za posamezno vsebino potrebuje le unikatno identifikacijo (CRID) [25] ter ocene posameznih uporabnikov, sam tip oziroma opis vsebine pa ga ne zanima. Slabost takega sistema je, da je zelo odvisen od števila uporabnikov, ki ta sistem uporabljajo. Če uporabnikov ni dovolj, potem sistem ni sposoben izbrati dovolj sosedov za aktivnega (trenutnega) uporabnika ter tako ni sposoben najti primernih vsebin.



Slika 8: Skupinsko filtriranje

Na sliki 8 je prikazen splošen sistem za skupinsko filtriranje. Pri gradnji uporabnikovega modela, sistem najprej iz baze uporabnikov poišče vse potencialne sosede aktivnega uporabnika. Le-te nato primerja z aktivnim uporabnikom na podlagi preteklih uporabnikovih dejanj (zabeleženih v obliki ocen vnesenih v bazi ocen) ter izbere najbolj primerne sosede. Ti sosedje se nato zabeležijo v uporabnikov model. Ko je potrebno uporabniku predlagati primerne vsebine, se uporabnikov model naloži v iskalnik primernih vsebin, ki na podlagi podatkov vnesenih v uporabnikovem modelu iz baze vsebin, ki so na voljo (torej vsebine, do katerih uporabnik še ni dostopal).

Pri skupinskem filtriranju pogosto nastopajo metoda K-najbližjih sosedov, metoda k-tih povprečij, metoda LBG ter korelacijske metode. Ker se v disertaciji osredotočimo na pristope iz teh skupin, bomo v sledečih podpoglavjih posamezne metode bolj natančno opisali.

2.4.2.1 Metoda K – najbližjih sosedov

Metoda k-najbližjih sosedov [10, 42] je primer metode, ki temelji na primerjavi profilov posameznih uporabnikov. Razvrščanje posameznih vsebin v skupine poteka preko določanja podobnosti med posameznimi uporabniki

Podobnost med uporabniki se izračuna na osnovi primerjave posameznih ocen, ki so jih uporabniki že vnesli v sistem. Rezultati primerjave posamezne ocene so nato združeni v oceno skupne podobnosti. Primer izračuna podobnosti med dvema uporabnikoma z dvajsetimi ocenami je prikazan z naslednjo enačbo:

$$sim(u_i, u_j) = \sqrt{\sum_{k=1}^{20} f(e(u_i, h_k), e(u_j, h_k))} \tag{2.1}$$

V enačbi nam u_i ter u_j predstavljata posameznega uporabnika. Ocena uporabnika u_i za *k*-to vsebino pa je zabeležena kot $e(u_i, h_k)$, kjer nam h_k predstavlja izbrano vsebino. Pomembno je opozoriti na to, da metoda primerja uporabnika samo glede na vsebine, ki sta jih ocenila oba. Če je prvi uporabnik ocenil vsebine, ki jih drugi še ni videl, se te ocene pri izračunu podobnosti ne upoštevajo (in obratno). Za primer smo izbrali, da uporabnika primerjamo na podlagi 20 skupnih vsebin, ker smo med našimi poskusi ugotovili, da je to najmanjše število vsebin, pri katerem sistem začne vračati uporabne rezultate.

Sam izračun podobnosti $sim(e(u_i, h_k), e(u_j, h_k))$ se lahko izvede na več različnih načinov. Nekatere možnosti so prikazane z naslednjimi enačbami:

$$sim(e(u_i, h_k), e(u_j, h_k)) = \begin{cases} 1 & e(u_i, h_k) \neq e(u_j, h_k) \\ 0 & e(u_i, h_k) = e(u_j, h_k) \end{cases}$$
$$sim(e(u_i, h_k), e(u_j, h_k)) = (e(u_i, h_k) - e(u_j, h_k))^2 \tag{2.2}$$

Sistem mora tako primerjati vsakega uporabnika z vsakim ter za vsak par uporabnikov izračunati mero podobnosti. Na koncu ima na voljo tabelo z vrednostmi podobnosti za vsako možno kombinacijo uporabnikov. Sistem je lahko zasnovan tako, da posodablja celotno

tabelo v enem koraku (torej, da res primerja vsakega z vsakim) ali pa samo podobnosti za izbranega uporabnika (torej osveži vrednosti podobnosti samo za pare uporabnikov, v katerih nastopa izbrani uporabnik). Ne glede na zasnovo je zadnji korak izbira K najbolj primernih sosedov za vsakega uporabnika. Za sosede se izbere uporabnike, ki se po izračunu podobnosti obravnavajo za najbolj podobne (točna številska vrednost je odvisna od izbrane metode izračuna). Ko imajo vsi uporabniki določene sosede, se postopek zaključi in sistem preide v fazo iskanja vsebin.

Za razvrščanje uporabnikov s to metodo potrebujemo več uporabnikov z vnesenimi ocenami. Metoda k-najbližjih sosedov dosega dobre rezultate v primeru, ko je število uporabnikov veliko ter ima vsak od uporabnikov vnešenih veliko ocen. Za razvrščanje posameznega uporabnika metoda zahteva primerjavo z vsemi ostalimi. Metoda je zato računsko zelo zahtevna in zato ni primerna za razvrščanje velike količine uporabnikov v kratkem času. Uporablja se tako pri vsebinskem kot pri skupinskem filtriranju vsebin.

2.4.2.2 Metoda k-tih povprečij

Metoda k-tih povprečij [42] je privlačna, ker jo je možno izvesti z uporabo preprostega in računsko nezahtevnega algoritma. Algoritem metode K-tih povprečij dodeli uporabnika tisti skupini, katere centroid (povprečje vseh točk, ki nam predstavlja središčno točko skupine) je najbližji uporabnikovemu profilu. Uporabnikov profil nam tu predstavljajo ocene vsebin, ki jih je vnesel v sistem. Podobnost med uporabnikom ter centroidom uporabniške skupine navadno izračunamo na podlagi Evklidove razdalje. Centroid je točka, katere vrednosti so izračunane kot povprečje vseh profilov uporabnikov, ki so dodeljeni v izbrano skupino. Algoritem se izvaja po naslednjih korakih:

i. Izberemo število rojev K.
ii. Naključno generiramo K rojev, izračunaj centroide ali neposredno generiraj K točk, ki služijo kot začetni centroidi skupin.
iii. Pripišemo vse vzorce najbližji skupni (glede na razdaljo do centroida).
iv. Ponovno izračunamo nove centroide.
v. Ponavljamo tretji ter četrti korak, dokler se centroidi skupin več bistveno ne spreminjajo.

Ko se centroidi ustalijo (torej da se središče centroida v sledečih korakih ne spreminja) in se metoda konča, so vsi uporabniki, ki so trenutno zabeleženi v sistemu, dodeljeni posameznemu centroidu. To storimo s pomočjo primerjave ocen, zabeleženih v uporabniških profilih s centrom vsakega posameznega centroida. Uporabnik je nato dodeljen centroidu, do katerega ima njegov profil najmanjšo (Evklidovo) razdaljo. Kot najbližji sosedi uporabnika se lahko nato obravnavajo vsi uporabniki, ki se nahajajo v istem centroidu. V primeru, da je v centroidu preveč uporabnikov, se lahko prostor ponovno razdeli na več (in tako manjših) centroidov ali pa znotraj posameznega centroida uporabimo dodatno metodo za izbiro najprimernejših sosedov (na primer metodo K – najbližjih sosedov, ki smo jo opisali v prejšnjem poglavju). Ko je vsak uporabnik dodeljen v centroid in ima izbrane sosede, lahko sistem preide v fazo iskanja primernih vsebin za posameznega uporabnika.

Slabost metode je, da moramo vnaprej poznati število iskanih skupin in da različna izbira začetnih točk lahko pripelje do različnih končnih rezultatov.

2.4.2.3 Metoda $N_C = 2^r$-tih povprečij

Metoda znana tudi kot LBG pristop (po avtorjih Linde, Buzo in Gray) [42] je bila osnovana za iskanje točno določenega števila skupin. Deluje na principu delitve skupin v podskupine (ang. *splitting method*). Metoda prostor iterativno razdeli na podskupine ter v vsaki iteraciji poveča število skupin. Potek metode:

i. Izračunamo centroid (povprečje vseh točk, ki nam predstavlja središčno točko skupine) vseh točk v prostoru.

ii. Vsak obstoječi centroid razdelimo na dva nova centroida, tako da prištejemo oziroma odštejemo pertubacijski vektor središču centroida.

iii. Za vsako točko v prostoru izračunamo razdaljo do posameznega centroida ter točko dodeli skupini, do katere ima najkrajšo razdaljo.

iv. Izračunamo nove centroide vsake obstoječe skupine.

v. Ponovimo od koraka ii naprej, dokler ne dosežemo želenega števila skupin.

Metoda deluje podobno kot metoda k-tih povprečij. Ko je prostor razdeljen na skupine, vsakega obstoječega uporabnika dodelimo v eno izmed skupin na podlagi primerjave profila ter izračunanega centroida vsake skupine (ponavadi z uporabo Evklidove razdalje). Za sosede uporabnika pa obravnavamo vse porabnike, ki se nahajajo v isti skupini. Ko je vsak obstoječ uporabnik dodeljen v skupino, lahko sistem preide v fazo iskanja primernih vsebin.

Slabost te metode (predvsem v primerjavi z metodo k-tih povprečij) je, da prostor razdeli na vnaprej določeno število skupin. Torej ni nujno, da dobimo optimalno število skupin, temveč bomo vedno dobili število skupin, ki je enako potenci števila 2. V zameno za to pa je metoda zelo hitra in nezahtevna.

2.4.2.4 Eigentaste

Eigentaste pristop je bil razvit na podlagi metode glavnih komponent (ang. principal component analysis – PCA) ter uporabljen v sistemu za iskanje primernih šal, opisanem v članku [43]. Vsakemu uporabniku sistem ob prvi prijavi ponudi za ocenjevanje točno določene vsebine. Na podlagi analize ocen teh vsebin uporabnika nato dodeli v eno izmed obstoječih skupin. Pristop je podrobneje opisan v poglavju 6.2, kjer smo ga preizkusili kot potencialni pristop za dodeljevanje uporabnikov iz naše podatkovne množice v skupine.

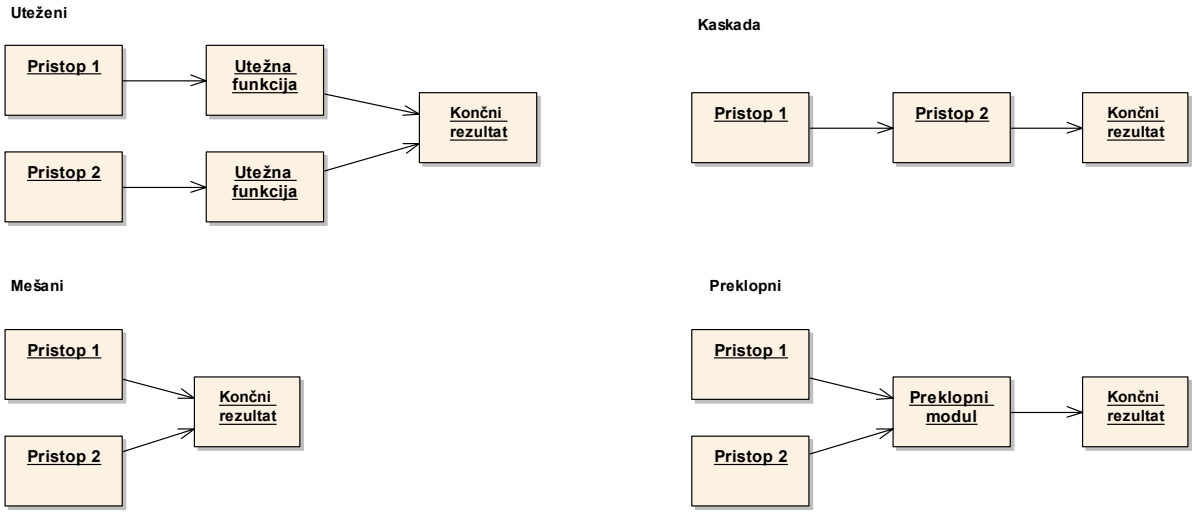
2.4.2.5 Korelacijska metoda (Pearsonov koeficient)

Za iskanje podobnih uporabnikov uporabljamo tudi korelacijske metode, ki z analizo izbranih ali ocenjenih vsebin skušajo določiti stopnjo podobnosti med uporabniki. Najpogosteje uporabljani postopek je Pearsonov korelacijski koeficient, ki so ga opisali v literaturi, povezani s projektom GroupLens [8]. Koeficient bomo podrobneje predstavili pri zasnovi sistema v poglavju 4.3.1.

2.4.3 Kombinirani pristopi

Izkazalo se je, da v splošnem noben pristop ne deluje optimalno, zato so se pojavili pristopi (ang. hybrid recommender), ki skušajo doseči boljše rezultate tako, da kombinirajo več različnih tehnik modeliranja [28, 44, 45]. Taki sistemi so znani kot hibridi oz. hibridni sistemi. Glavna prednost takih sistemov je, da so bolj prilagodljivi, in ker vsebujejo več pristopov, še vedno lahko vrnejo rezultat, tudi če eden od pristopov v dani situaciji odpove. Potencialna slabost hibridov podobno izhaja iz dejstva, da vsebujejo več pristopov – pri določenih zasnovah kombiniranih sistemov (na primer pri kaskadi) mora sistem upoštevati tudi zahteve vseh teh pristopov, kar lahko močno omeji možnosti uporabe sistema pri določenih kombinacijah. Tako npr. ne moremo imeti kombiniranega vsebinsko – skupinskega sistema, ki bi bil sposoben optimalno predlagati tako filme kot knjige, če vsebovani vsebinski pristop deluje samo z filmskimi vsebinami.

Glavna ideja kombiniranih pristopov niso novi pristopi, temveč različni načini kombiniranja obstoječih pristopov. Nekateri izmed njih so prikazani na sliki 9:



Slika. 9: Kombinirani pristopi

Opis posameznih pristopov, prikazanih na sliki:

- **Uteženi** (ang. *weighted*): pri uteženem kombiniranem sistemu vsak modul najprej neodvisno poišče primerne vsebine po lastnih kriterijih. V končni fazi pa nato modul kombiniranega sistema iz rezultatov posameznih modulov izbere vsebine, za katere je najbolj prepričan, da bodo pozitivno sprejete pri uporabniku.
- **Kaskada** (ang. *cascade*): zaporedni sistem deluje tako, da rezultate prvega modula uporabi kot vhodne podatke drugega ter tako naprej za vsak sledeči modul. Rezultati zadnjega modula pa se direktno ponudijo uporabniku.
- **Mešani** (ang. *mixed*): sistem uporabniku ponudi predlagane vsebine več različnih pristopov hkrati.
- **Preklopni** (ang. *switching*): izbere najbolj primernen pristop glede na situacijo (na primer koliko ocen je uporabnik že vnesel v sistem) ter nato s pomočjo tega pristopa poišče primerne vsebine.

2.4.4 Ostale skupine postopkov za filtriranje vsebin

Kot smo že omenili, ne obstajajo samo skupinske, vsebinske ter kombinirane skupine pristopov, temveč obstaja še veliko drugih. Izmed njih je vredno omeniti še [28]:

- **Demografski**: (ang. *Demographic*) pri demografskem pristopu sistem deluje na predpostavki, da imajo uporabniki s podobnimi demografskimi karakteristikami (starost, spol, življenjski status…) tudi podobne preference glede vsebin. Zato uporabnike dodeli v skupine s podobnimi karakteristikami in nato znotraj te skupine išče primerne vsebine za vsakega posameznega uporabnika. Za pravilno delovanje potrebujemo pri zasnovi sistema podatke o demografskih lastnostih ciljne skupine uporabnikov, katerim je sistem namenjen.
- **Sistem znanja**: (ang. *Knowledge-based*) tak sistem deluje podobno kot demografski. Zasnovan je na podlagi predznanja. Razlika je da knowledge-based pristop upošteva več različnih karakteristik uporabnikov. Uporablja ne samo demografske podatke, temveč tudi podatke o interesih, poslovnih zanimanjih, hobijih ter podobno. Pri zasnovi sistema morajo biti na voljo vsi ti podatki. Sistem je lahko boljši od osnovnih pristopov, vendar ima to slabost, da je v obstoječ sistem skoraj nemogoče dodati novo interesno skupino.

2.5 Pomankljivosti na področju uporabniku prilagojenega iskanja vsebin

Na področju uporabniku prilagojenega iskanja vsebin je prisotnih še veliko težav oziroma pomanjkljivosti, s katerimi se spopadajo raziskovalne skupine v okviru svojih raziskav [28, 44]. Nekatere od teh pomanjkljivosti so skupne vsem sistemom ne glede na pristop, druge pa so specifične za posamezno metodo.

2.5.1 Redkost podatkov

Težava, s katero se srečujejo vsi sistemi za uporabniku prilagojeno iskanje vsebin, je majhna količina podatkov, ki so na voljo. Ker so vsi sistemi mišljeni za uporabo v realnem okolju, to pomeni, da delujejo v okolju ki se dinamično spreminja – v sistem se vpisujejo novi uporabniki ter nove vsebine. Zato se baza podatkov, ki nastaja v okviru posameznega sistema, ves čas povečuje. Vendar ta baza nikoli ni polna oziroma je v večini primerov skoraj prazna, saj vsak uporabnik oceni zelo malo vsebin. To je pomanjkljivost, ki v resnici ni odpravljiva in jo je zato potrebno vedno upoštevati. Do tega pojava pride, ker realno ni možno, da bi vsi uporabniki videli ter ocenili vse obstoječe vsebine. Vsak uporabnik vidi zgolj zelo majhen del obstoječih vsebin ter za še manjši del od teh vsebin v sistem vnese povratno informacijo (oceno). Vse računske metode za analizo podatkovnih baz, ki so zasnovane za delo s polnimi matrikami, tako odpovejo in jih je potrebno prilagoditi ali zavreči.

2.5.2 Dodajanje novih uporabnikov

Pomankljivost, s katero se najbolj pogosto srečujejo uporabniku prilagojeni sistemi, se pojavi ob dodajanju novih uporabnikov v sistem. Ta pomankljivost se v literaturi pogosto obravnava z imenom hladni zagon (ang. Cold Start), ki se nanaša na uvajanje novosti v sistem. Le-ta namreč ne more delovati za novo prijavljenega uporabnika, če o njem še nima nobenega podatka oz. uporabnik v sistemu še ni ocenil nobene vsebine. Ena od predlaganih rešitev je na

primer anketa [43], ki bi jo uporabnik izpolnil ob prvem vstopu v sistem in tako dal vsaj začetne podatke, s pomočjo katerih bi sistem lahko začel pravilno funkcionirati. Prednost te rešitve je, da z izpolnjeno anketo sistem dobi uporabne podatke. Slabost te rešitve pa je, da zahteva določen napor od uporabnika na samem začetku, torej še preden bi videl, kakšne ugodnosti mu sistem sploh nudi. Podoben predlog je tudi uporaba privzetega profila ob prvem vstopu v sistem ter kasneje prilagajanje na podlagi uporabnikovih ocen. Veliko teh rešitev problem hladnega zagona omeji oziroma omili, splošne rešitve, ki bi ga popolnoma odpravila, pa še ni.

2.5.3 Vpeljava novih vsebin

Pomankljivost, podobna hladnemu zagonu, se pojavi tudi pri vpeljavi novih vsebin v sistem, ki deluje po principu vsebinskega filtriranja. Ker vsebinsko filtriranje deluje na podlagi analize metapodatkov vsebin, novih vsebin ne more pravilno upoštevati, če le-te niso opremljene s primernim opisom. Zato je potrebno vsako novo vsebino opremiti s primernim opisom, kar v primeru raznih multimedijskih vsebin ni enostavno. Trenutno še ni na voljo sistema, ki bi omogočal avtomatsko opisovanje vsebin, temveč mora to narediti uporabnik, ki vsebino želi dodati v sistem. To pa lahko pripelje do nepopolno opisanih vsebin in napak. Pomanjkljivost še dodatno ojača dejstvo, da na področju opisovanja vsebin ni prisotnega enotnega standarda (že v pričujočem delu smo opisali 3 različne standarde v poglavju 1.3.2). Zaradi tega je zasnova sistema za avtomatsko opisovanje vsebin še težja.

2.5.4 Prekomerno prileganje

Ko ima sistem na voljo vse potrebne podatke, še vedno lahko nastopijo težave. Ena izmed njih je pretirana prilagojenost uporabniškega profila. Če pride do te situacije, sistem uporabniku začne ponujati omejen nabor vsebin iz zelo ozkega (specifičnega) področja. Če na primer uporabnik na začetku nakaže, da so mu všeč nogometne tekme, lahko pride do situacija, kjer ga sistem zasipa samo s predlogi za tekoče nogometne tekme, nikakor pa ni sposoben ponuditi celovečernega filma na drugem kanalu, kajti zanj ne ve, da bi uporabniku ustrezal. Možne rešitve so vpeljava naključnosti v nabor predlaganih vsebin – da izmed desetih predlaganih vsebin eno ali dve popolnoma naključno predlagamo. Tako uporabniku ponudimo možnost, da izbira izven okvirja lastnega profila. Druga možna rešitev je tudi, da uporabniški model gradimo samo na podlagi **N** zadnjih povratnih informacij, ki jih je uporabnik dal sistemu, namesto da bi upoštevali vse. Na ta način je sicer uporabniški model malo bolj grobo izdelan, kot če bi upoštevali vse obstoječe ocene, vendar je hkrati osnovan na bolj svežih podatkih in tako mogoče bolj primeren.

2.5.5 Pomanjkanje standardov

Pomanjkanje standardov na področju uporabniku prilagojenega iskanja vsebin je velika težava, ki ni vezana neposredno na samo implementacijo sistema. Neodvisno poteka veliko vzporednih raziskav, zato prihaja do težav, ker vsaka skupina odkrije svoj pristop in ga začne uporabljati kot standard za svoje sisteme. Zato je težko primerjati ocene uspešnosti sistema, saj lahko ene skupine uporabljajo natančnost in delež pravilno najdenih vsebin (ang. Precision and Recall), druge NMAE, tretje pa razvijajo svoja merila uspešnosti. Problem je še veliko bolj opazen na področju sistemov, osnovanih na vsebinskem pristopu, kjer vlogo igra tudi sam opis vsebine. Na tem področju je že sedaj razvitih več standardov – Dublin Core, MPEG-

7, MPEG-21 ter TVAnytime – kar pomeni, da sistem, ki je osnovan na enem standardu, ne more delovati z vsebinami, opisanimi v drugem standardu.

Prav tako pomanjkanje standardov onemogoča prenosljivost uporabniških profilov med sistemi ter tako od uporabnika zahteva, da ima pravzaprav v vsakem sistemu drug (sistemu specifičen) profil. Redki uporabniki so pripravljeni imeti dvajset in več profilov ter vsakič na novo vnašati osebne ter druge podatke. Prihaja do poskusov uvedbe enotnega profila [46], vendar trenutno niso uspešni in rešitev še ni na vidiku.

3. Testno okolje

V tem poglavju bomo predstavili okolje, v katerem so potekali poskusi, podatke, ki so nam bili na voljo, in tehnike, s pomočjo kateri smo vrednotili uspešnost dobljenih rezultatov.

3.1 Testno okolje

Raziskave na področju uporabniku prilagojenega iskanja vsebin se prej ali slej znajdejo pred vprašanjem kako priti do testnega okolja, ki vsebuje zadosti uporabnikov ter vsebin. Ker sistem za pravilno delovanje potrebuje povratno informacijo s strani uporabnika, imajo razvijalci dve možnosti:

- Prva je, da poleg samega razvoja sistema poiščejo tudi začetno skupino uporabnikov, ki v testni fazi sodelujejo, dajejo mnenja o vsebinah in tako omogočijo pravilen razvoj. Slabost tega pristopa je, da nekateri (predvsem skupinski) pristopi zahtevajo večje število uporabnikov in večje število iteracij, preden začnejo pravilno delovati. Večina uporabnikov pa ni vedno pripravljena sodelovati oziroma je iskanje primerne skupine uporabnikov lahko prav tako dolgotrajen proces kot sam razvoj sistema.
- Druga možnost pa je uporaba že obstoječih podatkovnih množic, ki so na voljo raziskovalcem in razvijalcem. Prednost teh podatkovnih množic je ta, da vsebujejo veliko število uporabnikov in njihovih ocen ter tako nudijo dovolj veliko množico podatkov za popoln razvoj sistema. Slabost pa je, da gre za 'mrtve' podatke. Ko je sistem razvit in optimiziran s pomočjo teh podatkov, ni nujno, da bo tudi v realnih situacijah pravilno deloval. Zato je ponavadi potrebno tak sistem kasneje še prilagajati glede na ocene dejanskih uporabnikov.

V okviru doktorskega dela smo izbrali drugo možnost ter sistem osnovali na podlagi dveh podatkovnih množic, ki bosta podrobneje opisani v sledečih poglavjih.

Podatkovni množici sta bili hranjeni v SQL podatkovnih bazah, ki smo jih zaganjali v okviru programa XAMP [47]. Ta nam omogoča ustvarjanje novih podatkovnih baz, njih urejanje ter druge nastavitve.

Za razvojno okolje je bilo najprej potrebno izbrati programski jezik, v katerem bi sistem zasnovali. Odločili smo se za programski jezik Java [48]. Razlog za tako izbiro je bila možnost, da bi lahko kasneje z razvojem nadaljevali tudi v realnih sistemih. Java nam omogoča enostavno integracijo uporabniku prilagojenih storitev v obstoječe spletne strani s pomočjo javanskih programov (ang. *Applet*) ter spletnih storitev (ang. *Webservice*). Poleg tega je Java kot programsko okolje močno podprto tudi s strani mobilnih terminalov in nam ta izbira omogoča tudi kasnejši razvoj aplikacij na mobilnih terminalih. Programirali smo v programskem okolju Eclipse [49], ki je odprtokodni projekt, in ki uporabnikom omogoča hiter in natančen razvoj novih aplikacij.

Na začetku smo podatkovno bazo hranili na računalniku s procesorjem Celeron 1,7GHz s 512Mb delovnega pomnilnika. Na tem računalniku smo tudi razvijali ter preizkušali naše aplikacije. Zaradi programske zahtevnosti smo kasneje podatkovno bazo preselili na ločen računalnik z enakimi specifikacijami, kasneje pa smo aplikacije začeli razvijati na novem računalniku z 1.7 GHz CoreDuo dvojedrnim procesorjem ter 1Gb delovnega pomnilnika.

3.2 odatkovne množice

Za preizkušanje uspešnosti naših sistemom smo imeli na voljo dve močno različni podatkovni množici:

- EachMovie
- Siol

3.2.1 EachMovie

Podatkovna množica EachMovie je bil do nedavnega na voljo vsem raziskovalcem na področju uporabniškega modeliranja pod okriljem Digital Equipment Corporation [50]. Podatkovno množico so kasneje umaknili z uradne spletne strani, ko se je pojavila novejša, posodobljena množica imenovana MovieLens [51].

Podatkovna množica je zelo obsežna in vsebuje veliko število uporabnikov, vsebin in ocen. Predstavlja rezultat osemnajst mesecev zbiranja podatkov ter vsebuje 61265 uporabnikov, 1623 vsebin ter 2811718 ocen (torej je vnesenih 2.82% vseh možnih ocen). Podatki so združeni v treh datotekah:

person.txt:

- Id – enolična identifikacijska številka posameznega uporabnika
- Age – starost
- Gender – spol
- Zip_code – poštna številka stalnega prebivališča

vote.txt:

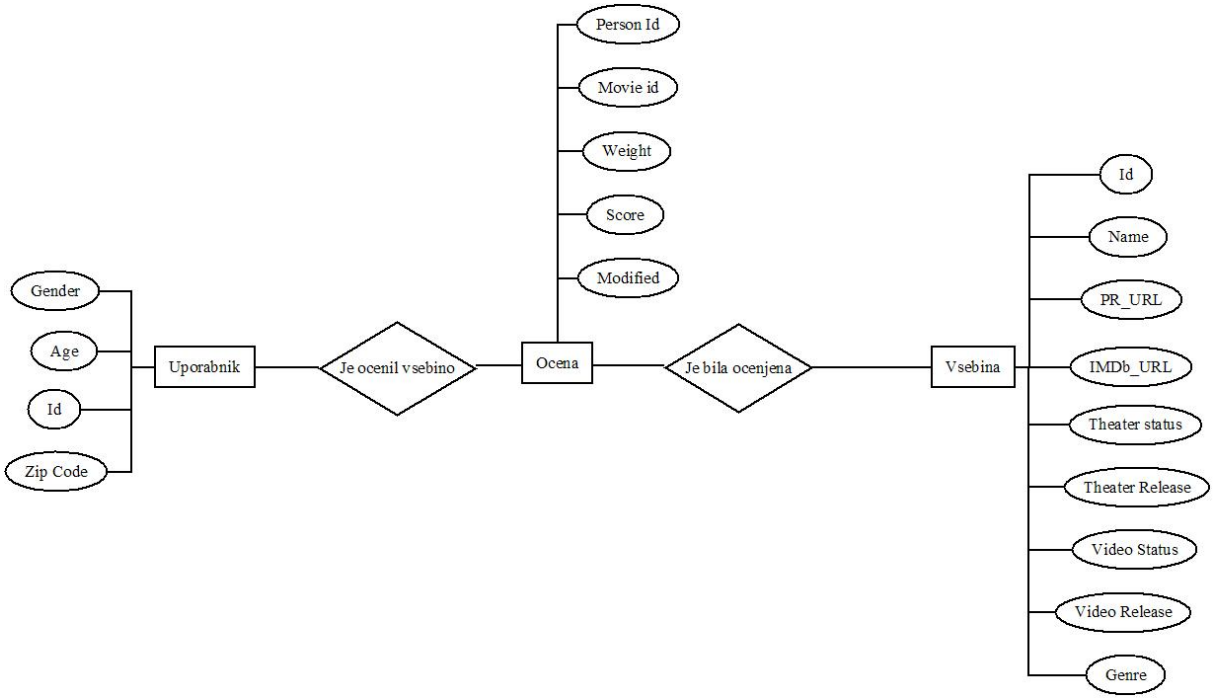
- Person_Id – identifikacijska številka uporabnika, ki je podal oceno
- Movie_Id – identifikacijska številka vsebine
- Score – ocena
- Weight – identifikacija načina podajanja ocene
- Modified – datum vnosa ocene

movie.txt:

- Id – identifikacijska številka vsebine
- Name – ime vsebine
- PR_URL – uradni spletni naslov vsebine
- IMDb_URL – spletni naslov vsebine na IMDB.com
- Theater_Status – podatek, ali gre za novo ali staro vsebino
- Theater_Release – datum, kdaj je bila vsebina predvajana v kinodvoranah
- Video_Status – podatek, ali gre za nov ali star video

- Video_Release – datum, kdaj je bila vsebina na voljo v video obliki
- Genre – podatek, o tem kateri žanri opisujejo to vsebino

Slika 10 prikazuje strukturo podatkovne baze Eachmovie z E-R diagramom.



Slika 10: E-R diagram EachMovie podatkovne baze

Podatkovna baza spada med večje, ki so na voljo raziskovalcem ter ima sledeče lastnosti:

- 61265 različnih uporabnikov,
- 1623 različnih vsebin,
- 2811718 vnesenih ocen,
- razpon ocen je od 0 (ni mi všeč) do 1 (zelo mi je všeč), v korakih po 0.2,
- povprečna ocena (izračunana na podlagi vseh obstoječih ocen) je 0.607,
- povprečna ocena posamezne vsebine je 0.502,
- povprečna ocena uporabnika je 0.643,
- povprečno število ocen za posamezno vsebino je 1732,42,
- povprečno število ocen, ki jih je uporabnik vnesel v bazo, je 45,89.

3.2.2 Siol

Drugo podatkovno množico smo sestavili s pomočjo podatkov o vsebinah, ki nam jih je zagotovil slovenski ponudnik internetnih IPTV storitev –SiOL [52]. SiOL svojim uporabnikom vsak dan nudi sporede za več kot 100 različnih kanalov, ki so uporabnikom dostopni preko IPTV omrežja.

Za testne namene smo dobili sporede vseh kanalov za obdobje dveh tednov v začetku leta 2007. To je približno 5000 TV-vsebin, ki pokrivajo širok spekter tipov vsebin. Za gradnjo podatkovne množice smo izmed teh vsebin izbrali 776 reprezentativnih vsebin, ki smo jih nato ponudili uporabnikom za ocenjevanje. Vsebine smo izbrali tako, da smo ohranili razmerje med tipi posameznih vsebin – če je tako v začetnih 5000 vsebin bilo 10% dokumentarnih oddaj, je tudi v naših 776 izbranih vsebinah vsebovanih 10% dokumentarnih oddaj.

Vsebine so bile opisane s SiOL-ovo EPG metapodatkovno strukturo, ki vsebuje kar 62 metapodtakovnih polj. Izbor najbolj zanimivih polj je prikazan v tabeli 2. Ker so bile naše raziskave osredotočene na skupinske pristope, nam večina teh polj ni bila zanimiva, edina izjema je bilo polje za opis kategorije, ki smo ga uporabili pri razvoju kombiniranega sistema, opisanega v šestem poglavju.

Ime Polja	Tip polja	Obvezno polje	Ime Polja	Tip polja	Obvezno polje
EventId	celo število	da	BlackAndWhite	boolean	ne
ProgrammeId	znakovni niz	da	Colorised	boolean	ne
ProgrammeName	znakovni niz	da	Category	znakovni niz	ne
StartDate	datum	da	Genre	znakovni niz	ne
StartTime	čas	da	Director	znakovni niz	ne
Duration	celo število	ne	Presenter	znakovni niz	ne
Title	znakovni niz	da	Actors	znakovni niz	ne
EpisodeTitle	znakovni niz	ne	Guests	znakovni niz	ne
EpisodeNumber	celo število	ne	CountryOfProduction	znakovni niz	ne
SeasonNumber	celo število	ne	Certificate	znakovni niz	ne
Synopsis	znakovni niz	ne	ProductionCompany	znakovni niz	ne
Description	znakovni niz	ne	YearOfProduction	celo število	ne
Language	znakovni niz	da			
Subtitles	boolean	ne			

Tabela 2: SIol EPG

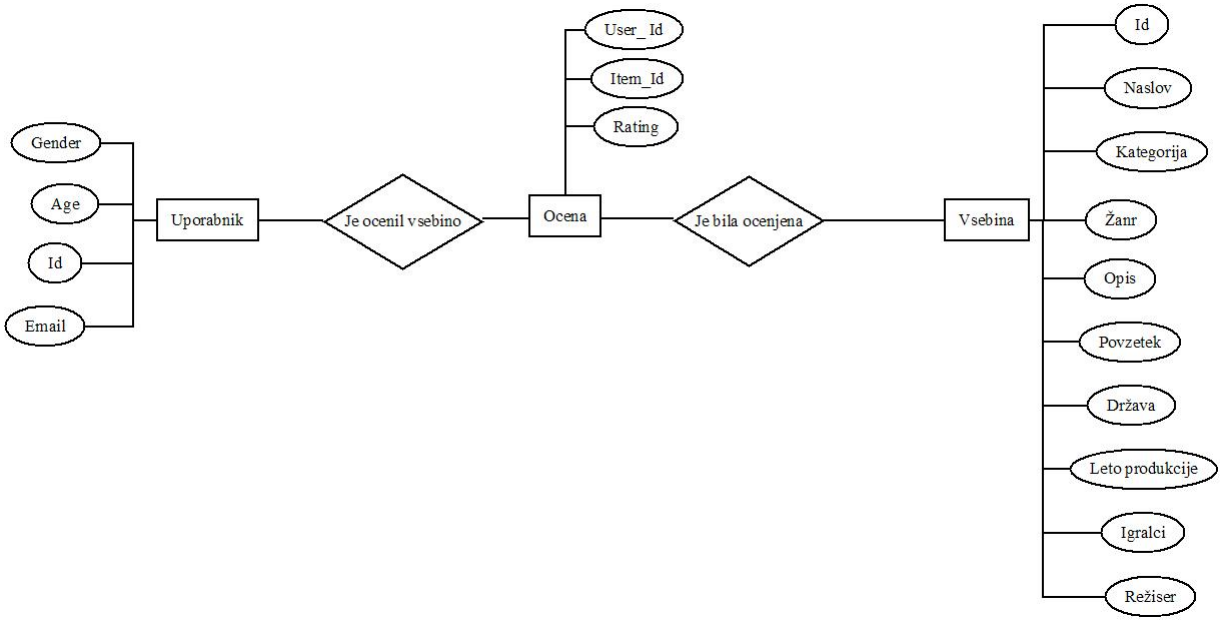
Za nadaljnje raziskave smo potrebovali tudi ocene za posamezne vsebine s strani čimvečjega števila uporabnikov, zato smo to podatkovno množico dali na voljo testnim uporabnikom. Vsebine je ocenjevalo 108 testnih uporabnikov, od tega 54 moških in 51 žensk, trije

uporabniki pa nam niso zaupali spola. Povprečna starost uporabnikov je bila več kot 23 let, uporabniki pa so bili stari med 17 in 57 let.

V testnem obdobju smo zbrali 12726 ocen uporabnikov. V povprečju je vsak uporabnik ocenil 110 vsebin.

Tako dobljena podatkovna množico je veliko manjša kot naša prva – EachMovie podatkovna množica. Vendar je izredno koristna predvsem zato, ker je veliko bolj polna kot prvotna podatkovna množica ter omogoča testiranje, kako se sistem obnese, ko ima na voljo večjo količino podatkov o okusu posameznega uporabnika, zato lahko uporabnike med seboj primerja po več ocenah hkrati.

Slika 11 prikazuje strukturo podatkovne baze Eachmovie z E-R diagramom.



Slika 11: E-R diagram Siol podatkovno bazo

Podatkovna baza je po obsegu manjša od EachMovie podatkovne baze, vendar vsebuje več ocen na posameznega uporabnika in zato nudi zelo drugačne razmere za testiranje (EachMovie baza vsebuje samo 2.8% možnih ocen, Siol pa 15.2%). Lastnosti baze so:

- 108 različnih uporabnikov,
- 776 različnih vsebin,
- 12726 vnesenih ocen,
- razpon ocen je od -2 (mi ni všeč) do 2 (mi je zelo všeč), v korakih po 1,
- povprečna ocena (izračunana na podlagi vseh obstoječih ocen) je 0.155,
- povprečna ocena posamezne vsebine je 0.4366,
- povprečna ocena uporabnika je 0.500,
- povprečno število ocen za posamezno vsebino je 16,39,
- povprečno število ocen, ki jih je uporabnik vnesel v bazo, je 113,625.

3.3 Postopek testiranja

Testiranje sistemov za uporabniku prilagojeno iskanje vsebin poteka v zadnji fazi uporabe sistema, torej takrat, ko sistem uporabniku predlaga vsebine, za katere misli, da bi ga zanimale. Preden lahko merimo uspešnost sistema, je torej potrebno sistem najprej naučiti o željah posameznih uporabnikov in mu omogočiti, da za vsakega uporabnika v podatkovni množici sestavi uporabniški model.

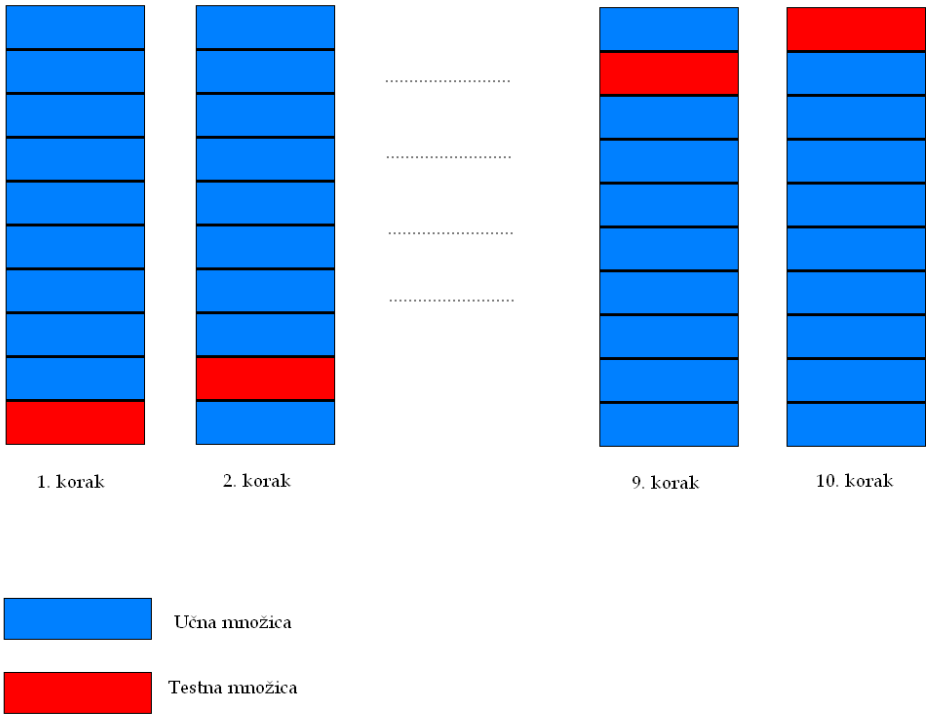
Razvoj sistema poteka v dveh fazah:

Faza učenja: v tej fazi se sistem uči o karakteristikah posameznega uporabnika na podlagi odkritih lastnosti zgradi uporabniške profile za vse uporabnike v obstoječi podatkovni množici.

Faza testiranja: sistem za vsakega uporabnika na podlagi njegovega profila poišče primerne vsebine ter izračuna napovedano oceno.

Da sistem pravilno deluje v teh dveh fazah, mora imeti na voljo dve podatkovni množici – učno ter testno množico (ang. *train set and test set*). Učno množico uporabi v fazi učenja, da iz nje razbere uporabnikova pretekla dejanja ter jih uporabi za gradnjo uporabnikovega profila. Testna množica pa se uporabi v fazi testiranja za primerjavo izračunanih (napovedanih) ocen sistema z uporabnikovimi dejanskimi ocenami, ki so zabeležene v testni množici.

Rezultate iz druge faze nato uporabimo za vrednotenje uspešnosti sistema. Pri sistemu, ki deluje z bazo aktivnih uporabnikov (torej z resničnimi uporabniki, ki se pogosto prijavijo v sistem in podajajo ocene), merimo uspešnost tako, da primerjamo napovedi sistema z resničnimi odzivi uporabnikov. Vendar pa pri razvoju novih sistemov za uporabniku prilagojeno iskanje vsebin to pogosto ni možno, predvsem zaradi dolgotrajnosti razvoja. To težavo rešujemo z uporabo že obstoječih podatkovnih množic, vendar je pri tem potrebna še dodatna obdelava njihovih podatkov. Če celotno podatkovno množico uporabimo za izgradnjo uporabniških modelov, sistema ni možno testirati v drugi fazi, saj nimamo več na voljo neodvisnih podatkov. Zato se v praksi uporablja metoda, imenovana prečno preverjanje (ang. ***n**-fold cross validation*) [2]. Pri tej metodi podatkovno množico razdelimo na **n** enako velikih podskupin. Eno izmed njih uporabimo za testiranje, vse ostale pa za gradnjo uporabniškega modela. Postopek nato **n**-krat ponovimo, da dosežemo, da se vsaka podskupina enkrat uporabi za testiranje sistema. Na ta način dobimo iz ene podatkovne množice **n** različnih podmnožic in lahko sistem večkrat preizkusimo. Primer 10-kratnega prečenega preverjanja je prikazan na sliki 12.



Slika 12: Primer 10-kratnega prečnega preverjanja

Ne glede na to, kateri pristop uporabimo, pri razvoju sistema vedno simuliramo isti scenarij. V praksi si prvo množico podatkov (torej učno množico) predstavljamo kot zabeleženo zgodovino uporabnikovih dejanj, ki bi jih imel sistem na voljo ob gradnji uporabnikovega profila. Na podlagi te zgodovine sestavimo profil in poiščemo primerne vsebine. Nato te vsebine 'ponudimo' uporabniku in zabeležimo njegov odziv. Uporabnikov odziv nam predstavlja ocena, zabeležena v testni množici, saj so zapisane ocene tam tudi resnični odzivi uporabnikov na izbrane vsebine. Pri delu z uporabniškimi sistemi ponavadi simuliramo eno samo interakcijo uporabnika s sistemom, oziroma kako se bo sistem obnesel s podatki, ki so mu na voljo. Sistem, ki bi zaporedoma simuliral več zaporednih interakcij med uporabnikom in sistemom za prilagojeno iskanje vsebin, pa trenutno še ne obstaja.

3.4 Mera učinkovitosti

Na področju preverjanja učinkovitosti sistemov za uporabniku prilagojenega iskanja vsebin se uporablja več meril. Za oceno uspešnosti našega sistema smo morali izbrati primerno metodo.

3.4.1 Delež najdenih, natančnost in mera F

V člankih o sistemih za uporabniku prilagojeno iskanje vsebin, se za vrednotenje uspešnosti zelo pogosto uporabljata merili za natančnost (ang. *Precision*) in delež pravilno najdenih vsebin (ang. *Recall*) ter njuna kombinacija mera F (ang. *F-measure*) [36]. Uporaba teh mer zahteva, da sledimo napovedanim in dejanskim ocenam za vsako vsebino, podobno kot pri prejšnjih metodah. Zahteva pa tudi, da vemo, katere vsebine bi sistem na koncu predlagal in katerih ne. Pri vrednotenju sistema torej upoštevamo tudi, kaj naj bi na koncu uporabnik res dobil in kaj ne. Pri tem uporabljamo sledeče vrednosti:

- ***Pravilno sprejeto*** (ang. *True positive* - TP) nam predstavlja število vsebin, za katere je sistem pravilno menil, da jih bo aktivni uporabnik pravilno sprejel ter jih zato uporabniku tudi ponudil v končnem izboru.
- ***Pravilno zavrnjeno*** (ang. *True negative* TN) pomeni število vsebin, za katere je sistem pravilno ugotovil, da jih uporabnik ne bo pozitivno sprejel ter jih je zato umaknil iz končnega izbora.
- ***Napačno sprejeto*** (ang. *False positive* FP) predstavlja vsebine, za katere je sistem zmotno pričakoval pozitivno reakcijo uporabnika ter jih zato vključil v končni izbor.
- ***Napačno zavrnjeno*** (ang. *False negative* FN) pa predstavlja vsebine, ki jih je sistem zmotno zavrgel, uporabnik pa bi jih v resnici pozitivno sprejel. Za določanje vrednosti teh kategorij je potrebno predhodno določiti, kaj naj sistem obravnava kot pozitivno reakcijo ter česa ne.

Vrednosti se računajo na podlagi primerjave vrednosti napovedane ocene $\hat{e}(u_a, h_p)$ ter dejanske ocene $e(u_a, h_p)$ z mejo, ki predstavlja pozitivno reakcijo, kjer je u_a profil aktivnega uporabnika, h_p pa predstavlja profil izbrane vsebine. V našem primeru smo imeli na voljo ocene v razponu od 0 (izjemno negativna reakcija) do 1 (izjemno pozitivna reakcija). Mejo za pozitivno reakcijo smo izbrali tako, da za pozitivno reakcijo sprejmemo samo zgornjih 30% ocen, torej razpon od 0.7 do 1 (Izbrano na podlagi ugotovitev v [53]). Vse ostalo pa obravnavamo kot negativno reakcijo. Izračun posamezne vrednosti je razviden iz sledeče tabele:

TP: $\hat{e}(u_a, h_p) > 0.7$ in $e(u_a, h_p) > 0.7$	TN: $\hat{e}(u_a, h_p) < 0.7$ in $e(u_a, h_p) < 0.7$
FP: $\hat{e}(u_a, h_p) > 0.7$ in $e(u_a, h_p) < 0.7$	FN: $\hat{e}(u_a, h_p) < 0.7$ in $e(u_a, h_p) > 0.7$

Tabela 3: Dodeljevanje v matriko razvrščanja

Ko so te vrednosti določene, lahko z njimi izračunamo natančnost – P in delež pravilno najdenih vsebin – R po sledečih formulah [36]:

$$P = \frac{TP}{TP + FP} \quad R = \frac{TP}{TP + FN} \tag{3.1}$$

Izračunani vrednosti nam nazorno povesta, kako se sistem obnese. Lahko zavzameta vrednosti od 0 do 1. Če delež pravilno razvrščenih vsebin zavzame vrednost 1, to pomeni, da sistem vse vrednosti, ki jih izbere za potencialno zanimive, tudi pravilno razporedi in ne predlaga napačnih. Če delež pravilno najdenih zavzame vrednost 1, pa pomeni, da sistem pravilno predlaga vse vsebine ter nobene ne izpusti po pomoti.

Za končno predstavitev uspešnosti sistema se uporablja še vrednost mera F (označimo z FM), ki vrednosti obeh deležev združi v eno samo številko. Izračunamo jo kot:

$$FM = \frac{2PR}{P + R} \tag{3.2}$$

Izračunana mera nam poda oceno, kako dobro se sistem obnese. Zavzame lahko vrednost od 0 do 1. Ponovno rečemo, da sistem z vrednostjo mere F enako 1 za vse vsebine pravilno ugotovi, ali so primerne za uporabnika.

Vrednost mere F naključnega klasifikatorja je odvisna od same strukture testne množice. V primeru naših učnih množic se tako ta vrednost spreminja od eksperimenta do eksperimenta. Vrednost za celotno učno množico je 0,4752. Zaradi primerljivosti smo primerjalno mejo postavili na F = 0,5 za vse eksperimente, kar je na varni strani za primerjavo.

Poleg tega je za nas zanmiva tudi vrednost mere F = 0.6. Ta meja predstavlja rezultate naših začetnih poskusov, objavljenih v [54], kjer smo testirali prototip sistema. Naš cilj je to mero čimprej preseči in tako sistem še dodatno izboljšati.

3.4.2 Statistično testiranje

Poleg grafične predstavitve rezultatov s prikazom poteka mere F, smo želeli tudi statistično preveriti, če so razlike prikazane na grafih resnične ali zgolj rezultat porazdelitve naših vzorcev v testni množici. Da bi to lahko ugotovili je potrebno izvesti statistično testiranje oziroma preveriti ničelno hipotezo. Testiranje hipoteze smo opravili povsod kjer smo primerjali metode med seboj (poglavja 4.5.1, 5.1.1, 5.1.2, 5.2 in 6.7).

Testiranje hipoteze smo najprej želeli izvesti na podlagi t-testa[55] vendar se je izkazalo, da pogoji testa niso izpolnjeni (porazdelitve v primerjanih grupah niso normalne, kar smo preverili s Kolmogorov-Smirnov testom [56]). Zato smo izbrali nadomestni statistično šibkejši test – Mann-Withney test, s katerim testiramo isto aplikativno hipotezo [57]. Zanj so pogoji testiranja izpolnjeni. Vse teste smo izvedli pri stopnji tveganja α=0,05.

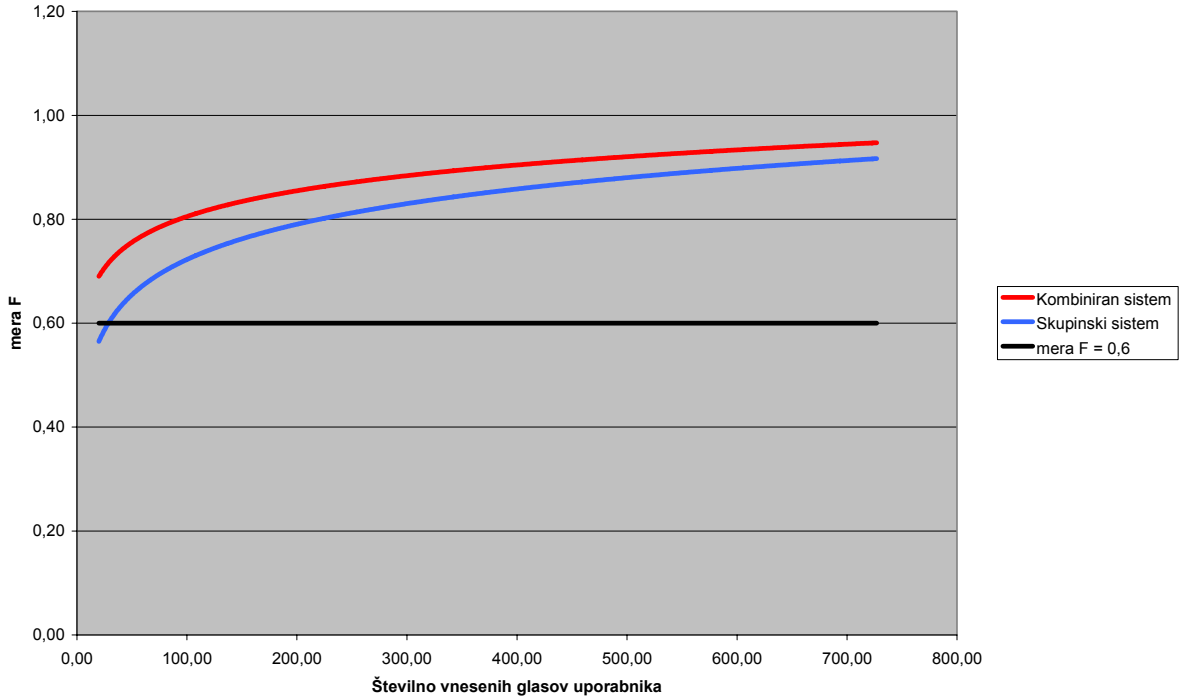
V dodatku B na strani 99 so prikazani rezultati vseh izvedenih Man-Withney testov. Vsakega izmed potekov smo primerjali ob različnih številih vnesenih glasov, kar predstavlja obnašanje sistema ob različnih časovnih obdobjih (torej kmalu po prvi prijavi uporabnika v sistem, kasneje ko začne uporabnik sistem aktivno uporabljati ter, veliko kasneje, ko je uporabnik v sistem vnesel že veliko število glasov). Ničelna hipoteza testa je bila da dva vzorca izhajata iz iste populacije vzorcev. S pomočjo te hipoteze smo želeli ugotoviti ali se rezultati dveh metod med seboj signifikantno razlikujejo. Rezultate teh testov komentiramo v poglavjih na katere se nanašajo.

3.5 Predstavitev rezultatov

Rezultate bomo predstavljali grafično, ker želimo čim bolj nazorno prikazati, kako se zanesljivost sistema boljša ali slabša z naraščanjem števila glasov posameznega uporabnika. To je glavni način za oceno primernosti sistemov za uporabniku prilagojeno iskanje vsebin in se tudi uporablja v večini strokovnih objav na tem področju.

Podajanje zanesljivosti v odvisnosti od števila zabeleženih glasov je najbolj verodostojno merilo, saj nam omogoča, da ocenimo, kako bo sistem napredoval z uporabnikom, torej kako uspešno se mu bo prilagajal. Z vsako vneseno oceno namreč uporabnik natančneje izrazi svoje preference, in če sistem deluje pravilno, potem z vsako tako oceno tudi zanesljiveje najde primerne vsebine.

Primer podajanja rezultatov je razviden iz slike 13.



Slika 13: Primer podjanja rezultatov

Na vseh slikah bo poleg poteka rezultatov za lažje prepoznavanje kdaj sistem začne kvalitetno iskati primerne vsebine za uporabnika, označena tudi meja, ki predstavlja vrednost mere F enako 0.6 (na sliki označeno s črno črto).

4. Razvoj sistema za skupinsko filtriranje

Znanstvena literatura s področja pristopov skupinskega filtriranja vsebin navaja, da se pri razvoju novih sistemov skupinskega filtriranja zelo pogosto uporablja Pearsonov korelacijski koeficient [8] za iskanje podobnih uporabnikov in utežena vsota za iskanje primernih vsebin. Odločili smo se preizkusiti uspešnost teh pristopov na podatkovnih množicah, ki so nam bili na voljo in na podlagi ugotovitev razviti nov sistem za skupinsko filtriranje.

V prvem koraku naših raziskav smo preizkusili Pearsonov korelacijski kofecient kot metodo za iskanje najbližjih sosedov in identifikacijo prednosti in slabosti le-tega. Pristop smo uporabili v našem razvojnem okolju in preizkusili s pomočjo podatkovnih množic.

V drugem koraku smo razvili novo metodo za skupinsko filtriranje vsebin, ki vsebuje več pristopov in lahko preklaplja med njimi. Ugotovili smo namreč, da noben pristop ni optimalna rešitev za vsako možno situacijo, zato smo želeli ugotoviti, če lahko s kombiniranjem več pristopov izboljšamo uspešnost sistema.

V sledečih poglavjih opisujemo posamezne korake raziskave, rezultate in zaključke, do katerih smo v teh korakih prišli.

4.1 Potek skupinskega modeliranja

Za boljše razumevanje razvitega sistema za skupinsko filtriranje moramo najprej predstaviti osnovno delovanje splošnega skupinskega sistema. Proces skupinskega filtriranja je sestavljen iz dveh ločenih faz. V prvi fazi sistem spoznava uporabnika in na podlagi njegovih ocen sestavi njegov uporabniški model. V drugi fazi pa nato ta model uporabi za iskanje primernih vsebin, ki jih nato ponudi uporabniku.

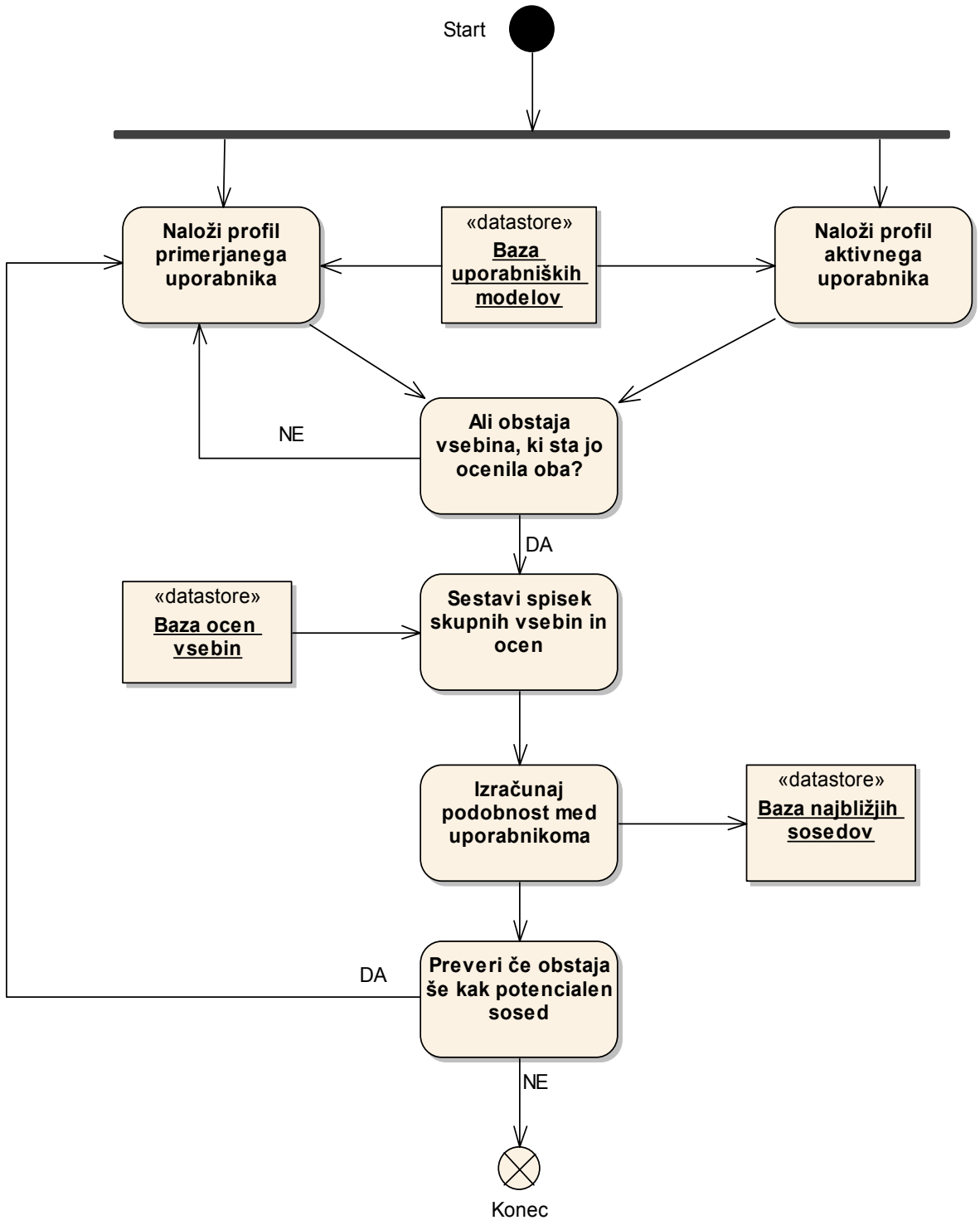
4.1.1 Gradnja uporabniškega modela

Gradnja uporabniškega modela je prva faza sistema za uporabniku prilagojeno iskanje vsebin. Zato da lahko sistem uporabniku išče primerne vsebine, mora najprej analizirati njegov okus oziroma njegove preference [1, 2, 17]. V modelu se shranjujejo vrednosti spremenljivk metode, s pomočjo katere sistem kasneje išče primerne vsebine. Tip in vrednosti spremenljivk so odvisne od uporabljenih metod. Ne glede na izbran pristop model vedno vsebuje spisek uporabnikov, ki imajo podoben okus kot lastnik trenutnega profila. V tej fazi torej sistem vsakemu obstoječemu uporabniku $u \in U$ zgradi njegov uporabniški model $um(u) \in UM$, kar označimo s preslikavo $um: U \rightarrow UM$ (glej 2.2).

Glavna naloga sistema v tej fazi je iskanje uporabnikov s podobnim okusom – najbližjih sosedov. Ko sistem ugotovi katero metodo bo uporabil, lahko začne z iskanjem sosedov. Kot potencialni sosed se obravnava vsak obstoječi uporabnik v podatkovni bazi, ki je v sistem vnesel vsaj nekaj ocen. To pomeni, da mora sistem naložiti podatke o vsakem uporabniku, ga primerjati z aktivnim uporabnikom in izračunati, kako primeren je za soseda. Računanje primernosti sosedov poteka na podlagi primerjave ocen za vsebine, ki sta jih oba uporabnika ocenila. Če je aktivni uporabnik ocenil na primer 100 vsebin, potencialni sosed pa je od teh stotih vsebin ocenil samo dvajset, bo primerjava potekala na podlagi teh dvajsetih vsebin. Sistem torej za vsakega potencialnega najbližjega soseda $u_{ps} \in U$ izračuna njegovo primernost (podobnost) $sim(u_a, u_{ps})$ s trenutnim aktivnim uporabnikom $u_a \in U$ (uporabnik,

ki mu želimo določiti najbližje sosede). Na podalgi izračunane primernosti nato uporabnika u_{ps} sprejme kot soseda ali pa ga zavrže. (psevdokoda postopka se nahaja v dodatku A.2)

Za ilustracijo navajamo primer primerjave dveh uporabnikov.



Slika 14: Primerjava uporabnikov

Za vsakega od uporabnikov imamo na voljo njegov profil $um(u_i)$ oz $um(u_j)$, v katerem so shranjene ocene, ki jih je do sedaj vnesel v sistem. Ocene zapisujemo v obliki $e(u_i,h_k)$, kjer nam e predstavlja dejansko vrednost ocene, u_i predstavlja uporabnika, ki je vsebino ocenjeval, h_k pa vsebino, ki je bila ocenjevana. Sistem mora najprej ugotoviti, v katerih ocenah se uporabnika prekrivata, torej ali obstaja $e(u_i,h_k)$ ter $e(u_j,h_k)$. Uporabnika bomo namreč lahko primerjali samo, kadar imamo na voljo obe vrednosti. Ko sistem ugotovi, katere pare ocen ima na voljo na podlagi enega izmed postopkov, opisanih v poglavju 2.4.2, izračuna vrednost $sim(u_i, u_j)$, ki nam predstavlja številsko mero podobnosti med obema uporabnikoma. Interpretacija te vrednosti je odvisna od uporabljene metode.

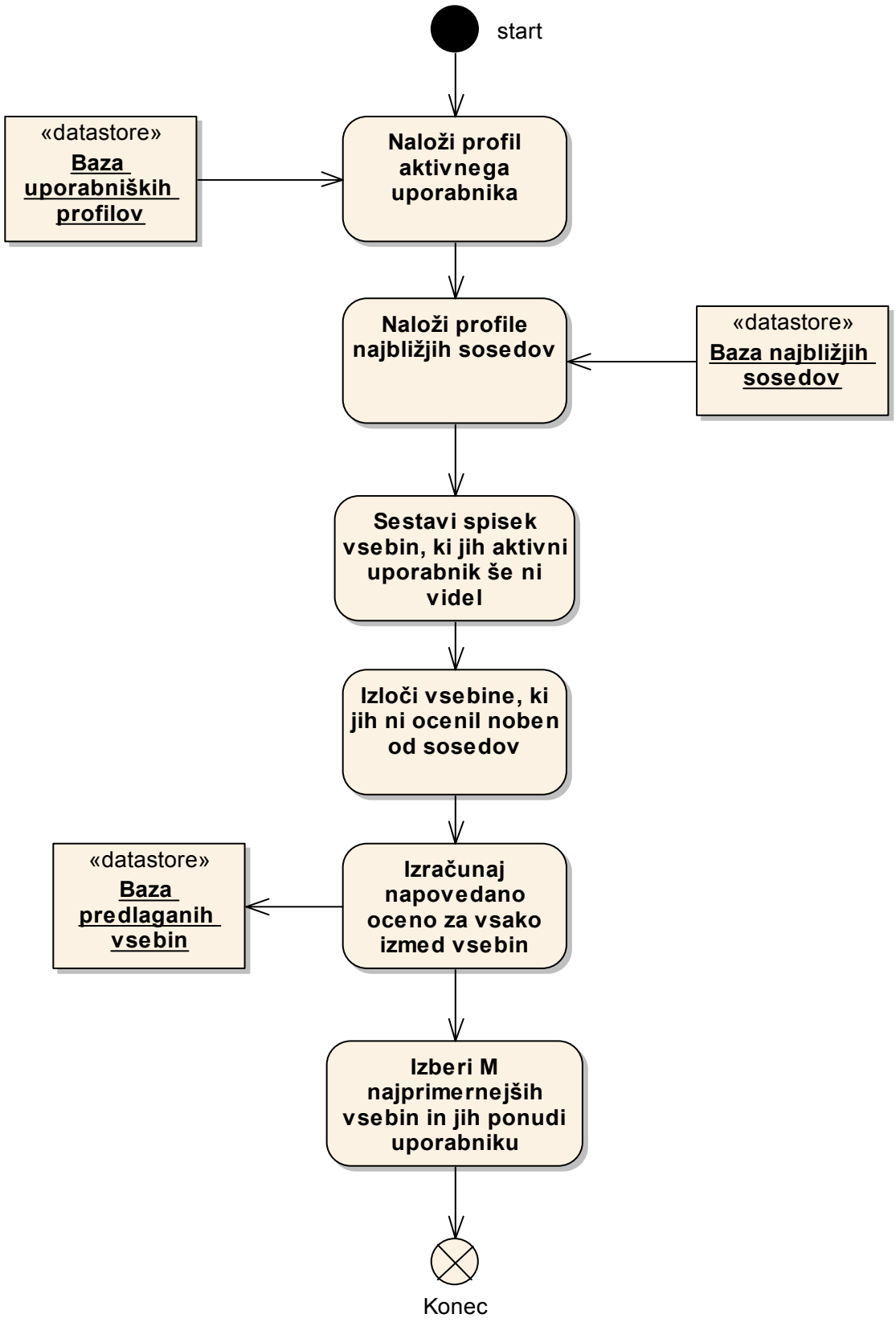
Rezultat postopka na sliki 9 je tabela, ki vsebuje identifikacije uporabnikov-sosedov in izračunane podobnosti z aktivnim uporabnikom. Ko sistem primerja vse obstoječe uporabnike z aktivnim uporabnikom, mora iz te tabele izbrati najbolj primerne uporabnike in jih zabeležiti kot najbližje sosede aktivnega uporabnika. Število sosedov, ki so vsebovani v končnem uporabnikovem profilu, je odvisno od izbranega pristopa za iskanje primernih vsebin (kot je podrobneje razloženo v poglavju 5.1). Ko so najbližji sosedi aktivnega uporabnika znani, lahko sistem preide v drugo fazo – iskanje primernih vsebin.

4.1.2 Iskanje primernih vsebin

Sistem za skupinsko filtriranje vsebin išče primerne vsebine na osnovi podatkov, katerih uporabniki so bili označeni za sosede aktivnega uporabnika. Ko te podatke pozna, mora najprej sestaviti množico vsebin, ki bi bile potencialno zanimive za aktivnega uporabnika. Kot potencialno zanimive vsebine obravnava vse vsebine, ki so jih sosedje ocenili, aktivni uporabnik pa še ni dostopal do njih. Ko so potencialno zanimive vsebine znane, je naloga sistema določiti, katere izmed njih so bile sosedom najbolj všeč ter jih nato ponuditi aktivnemu uporabniku. Sistem torej deluje na predpostavki, da so vsebine, ki so všeč sosedom, zaradi podobnosti okusov všeč tudi aktivnemu uporabniku.

Sistem ima torej na voljo podmnožico vsebin $H_p \subset H$, za katere ne obstaja ocena aktivnega uporabnika $e(u_a,h_p)$, obstaja pa vsaj ena ocena najbližjih sosedov $e(u_s,h_p)$. Sistem nato za vse vsebine $h_p \in H_p$ izračuna napovedano oceno $\hat{e}(u_a,h_p)$ na podlagi preslikave $\hat{e}: H_p \times E \to \hat{E}$. Izračunana napovedana ocena $\hat{e}(u_a,h_p)$ predstavlja napoved, kako bo uporabnik vsebino ocenil, ko bo dostopal do nje. Postopek izračuna napovedane ocene mora ponoviti za vsako izmed potencialno zanimivih vsebin h_p (postopki izračuna napovedane ocene so podrobneje opisani v poglavju 4.4). V zaključku izmed vseh potencialnih vsebin izbere **M** vsebin, ki imajo najvišjo izračunano napovedano oceno. **M** je običajno manjša številka, ne večja od 10, saj uporabnik ne želi biti soočen s preveč predlogi. Glavni razlog za tako majhno število je ugotovitev, da je povprečen uporabnik sposoben hkrati obravnavati 7-9 vsebin [56]. Uporabnik bo izmed ponujenih vsebin izbral samo eno in podal svoje mnenje (oceno) o vsebini $e(u_a,h_p)$. V naslednjem koraku je potrebno vsebine ponovno iskati s pomočjo posodobljenega uporabnikovega modela. Cilj je torej uporabniku ponuditi nabor vsebin, ki je dovolj majhen, da ga lahko uporabnik brez prevelikega napora preuči in iz njega izbere vsebino, ki mu najbolj ustreza. V primeru zelo velikega nabora ponujenih vsebin pa bi uporabnik zelo hitro dobil vtis, da sistem ne nudi nobene uporabne funkcionalnosti.

Potek iskanja primernih vsebin je prikazana na sliki 15, psevdokoda pa se nahaja vdodatku A.3.



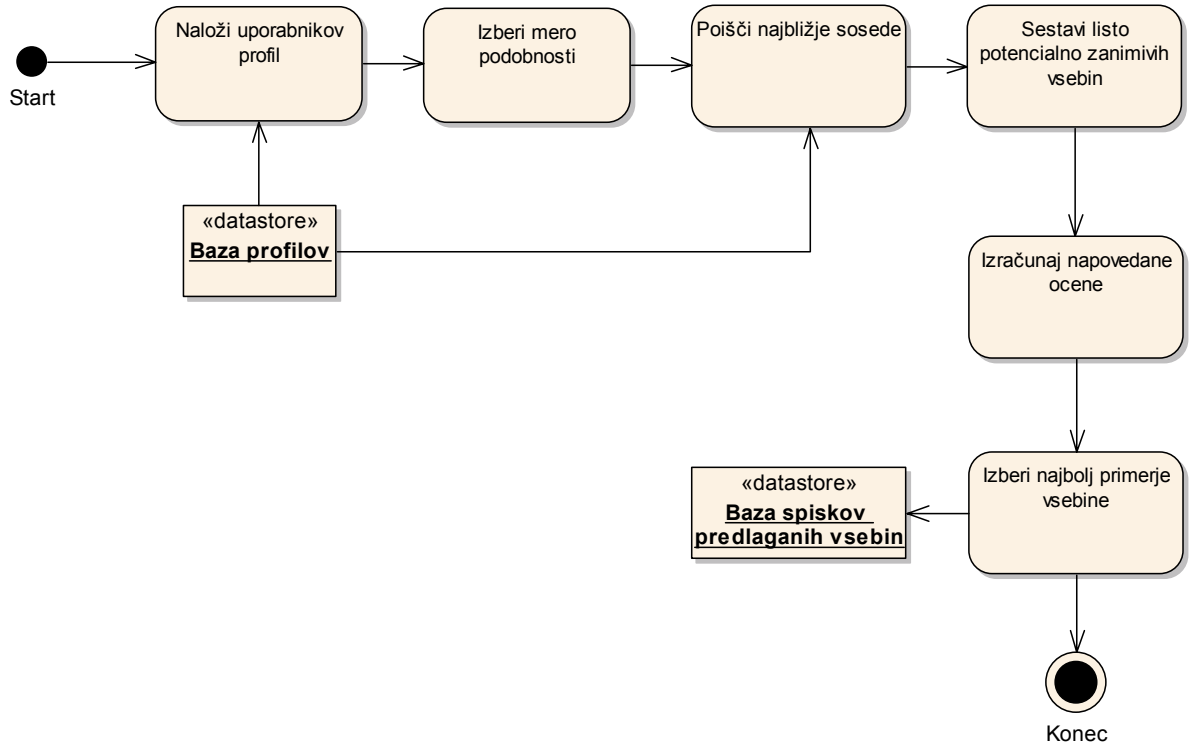
Slika 15: izbira primernih vsebin

4.2 Zasnova sistema

Sistem, ki smo ga razvili pri naših raziskavah, je bil osnovan v programskem okolju Java [48]. To programsko okolje smo izbrali iz več razlogov, glavni pa je enostavna prenosljivost med različnimi operacijskimi sistemi in mobilnimi napravi. Programsko okolje nudi tudi širok nabor predhodno sestavljenih programskih knjižnic, ki so nam olajšale delo s podatkovnimi bazami. Programirali smo v programskem okolju Eclipse [49], ki je na voljo kot odprtokodna aplikacija. Sistem je deloval na osebnem računalniku z operacijskim sistemom Windows XP Professional s procesorjem Celeron 2.8 GHz in 512 Mb pomnilnika.

Potrebovali smo tudi dostop do podatkovnih množic opisanih v poglavju 3.2. Podatkovne množice so bile hranjene v mySQL podatkovni bazi, ki se je na začetku nahajala na istem računalniku, kasneje pa smo jo preselili na ločen računalnik.

Celotni sistem je bil zasnovan v več ločenih javanskih objektih, ki so prikazani na sliki 16.



Slika 16: Tok izvajanja sistema za skupinsko filtriranje vsebin

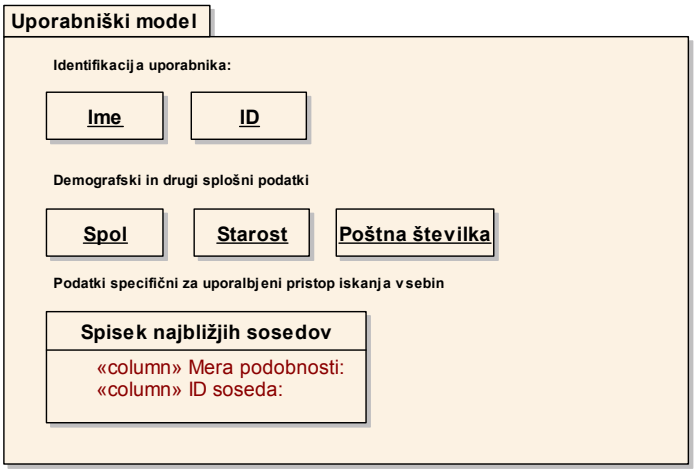
Program je zasnovan tako, da je ob zagonu izvedel vse operacije, potrebne za vsebinsko filtriranje vsebin – izgradnjo uporabniškega modela ter izbiro primernih vsebin. V sledečih poglavjih bomo podrobneje opisali postopke za iskanje najbližjih sosedov (poglavje 4.3) ter postopke za izbiro primernih vsebin (poglavje 4.4).

Potek celotnega postopka izgradnje uporabniškega modela je podan v dodatku s psevdokodo (glej str 84).

4.3 Izbira metode iskanja sosedov

Pri izbiri metode za iskanje najbližjih sosedov smo se najprej osredotočili na metodo, zasnovano na Pearsonovem korelacijskem koeficientu [8].

Uporabniški model, zasnovan z uporabo tega pristopa, je zelo enostaven in vsebuje zelo malo podatkov, kot je razvidno iz slike 17.



Slika 17: Primer vsebine uporabniškega modela

Poleg osnovnih informacij o uporabniku (spol, ime, priimek, starost in poštna številka) vsebuje samo eno pomembno podatkovno strukturo, s pomočjo katere sistem kasneje išče primerne vsebine. Ta podatkovna struktura je predstavljana kot tabela uporabnikov, za katere je sistem ocenil, da imajo najbolj podoben okus in jih je zato označil za najbližje sosede.

Pri iskanju sosedov se pri večini metod podobnost med dvema uporabnikoma $Sim(u_a, u_{ps})$ izračuna s pomočjo primerjanja ocen vsebin, ki sta jih ocenila oba uporabnika, kot smo opisali v poglavju 2.4.2. Rezultat tega izračuna je pogosto številska vrednost, ki nam predstavlja stopnjo podobnosti med uporabnikoma. Sistem nato izbere za najbližje sosede uporabnike, ki po izračunani podobnosti pridejo najbližje iskani vrednosti podobnosti.

Pri pregledu literature [42] vidimo, da imamo na voljo več različnih metod primerjave dveh vzorcev – uporabnikov.

Razdalje osnovane na p-normi:

Poznamo več različnih mer razdalje med dvema uporabnikoma, osnovanih na p-normi. Med njimi poudarjamo Evklidsko razdaljo ($p = 2$), Manhattan razdaljo ($p = 1$) ter razdaljo Čebiševa $p = \infty$.

Evklidova razdalja

Evklidova razdalja je edina razdalja, ki jo lahko izračunamo direktno s skalarnim produktom, ki je povezan s koreliranostjo podatkov. Izmed obstoječih p-norm nam predstavlja kompromis med upoštevanjem posamičnih odklonov in merjenjem velikega števila majhnih odklonov. Računamo jo po sledeči enačbi:

$$sim(u_a, u_j) = \sqrt{\sum_{k=1}^{n} (e(u_a, h_k) - e(u_j, h_k))^2} \qquad (4.1)$$

kjer je $sim(u_a, u_j)$ podobnost med aktivnim uporabnikom u_a in potencialnim sosedom u_j izračunana glede na vse vsebine h_k, ki sta jih ocenila oba uporabnika. Ocena za izbrano vsebino je predstavljena z $e(u_a, h_k)$ za aktivnega uporabnika in $e(u_j, h_k)$ za potencialnega soseda.

Manhahattan razdalja

Manhtan razdalja nam predstavlja p-normo, ki se osredotoči na merjenje velikega števila majhnih odklonov. Računamo jo po sledeči enačbi:

$$sim(u_a, u_j) = \sum_{k=1}^{n} \left| e(u_a, h_k) - e(u_j, h_k) \right| \tag{4.2}$$

kjer je $sim(u_a, u_j)$ podobnost med aktivnim uporabnikom u_a in potencialnim sosedom u_j izračunana glede na vse vsebine h_k, ki sta jih ocenila oba uporabnika. Ocena za izbrano vsebino je predstavljena z $e(u_a, h_k)$ za aktivnega uporabnika in $e(u_j, h_k)$ za potencialnega soseda

Ker se osredotoči na merjenje majhnega števila odklonov, je njena uporabnost za iskanje najbližjih sosedov vprašljiva, saj je lahko zelo pomemben tudi podatek, da si dva uporabnika pri oceni določene vsebine zelo močno nasprotujeta.

Čebiševa razdalja

Čebiševa razdalja v nasprotju z razdaljo Manhattan zanemarja posamezne manjše razlike med vzorci ter se namesto tega osredotoči na detekcijo posameznih velikih odklonov. Izračunamo jo po sledeči enačbi:

$$sim(u_a, u_j) = \max_{k=1\ldots n} \left\{ \left| e(u_a, h_k) - e(u_j, h_k) \right| \right\} \tag{4.3}$$

kjer je $sim(u_a, u_j)$ podobnost med aktivnim uporabnikom u_a in potencialnim sosedom u_j izračunana glede na vse vsebine h_k, ki sta jih ocenila oba uporabnika. Ocena za izbrano vsebino je predstavljena z $e(u_a, h_k)$ za aktivnega uporabnika in $e(u_j, h_k)$ za potencialnega soseda.

Mahalanobiseva razdalja

Računa podobnost med dvema vzorcema-uporabnikoma na podlagi lastnih vektorjev, torej v prostoru kjer upošteva korelacije med vzorci. Na ta način lahko zelo natančno izmerimo podobnost med vzorci, vendar je izračun te podobnosti računsko kompleksen (operacije z matrikami, predhodno iskanje lastnih vektorjev) in zato ne nujno primeren za sistem z zelo velikim številom vzorcev. Izračunamo jo po sledeči enačbi:

$$sim(u_a, u_j) = \left(u_a - u_j\right)^T \mathbf{K}^{-1} \left(u_a - u_j\right) \tag{4.4}$$

kjer $sim(u_a, u_j)$ predstavlja izračunano mero podobnosti, u_a, profil uporabnika, kateremu iščemo sosede, u_j pa profil potencialnega soseda. $\mathbf{K}$ je kovariančna matrika ocen uporabnikov.

Utežena razdalja Minkovskega

Razdalja pravzaprav predstavlja p-normo, pri kateri lahko posameznim značilkam vzorca dodelimo večji ali manjši pomen s primernimi vrednostmi uteži. Izračunamo jo po sledeči enačbi:

$$sim(u_a, u_j) = \left(\sum_{k=1}^{n} w_k \left| e(u_a, h_k) - e(u_j, h_k) \right|^p \right)^{\frac{1}{p}} \tag{4.5}$$

kjer je $sim(u_a, u_j, h_k)$ podobnost med aktivnim uporabnikom u_a in potencialnim sosedom u_j izračunana glede na vse vsebine h_k, ki sta jih ocenila oba uporabnika. Ocena za izbrano vsebino je predstavljena z $e(u_a, h_k)$ za aktivnega uporabnika in $e(u_j, h_k)$ za potencialnega soseda. Vrednost w_k pa predstavlja utež, s katero ovrednotimo pomembnost posamezne ocene.

Ker nam pri sistemih za skupinsko filtriranje vsebin vrednosti značilk vzorca predstavljajo uporabnikove ocene za posamezne vsebine ter vse vsebine obravnavamo enakovredno, ta razdalja ni primerna, saj bi vse uteži morale biti enake vrednosti.

Normiran korelacijski koeficient ali Pearsonov korelacijski koeficient

Korelacijski koeficient meri linerni del povezanosti med dvema uporabnikoma – vzorcema. Če med vzorci ne obstaja linerna povezanost, se metoda slabo obnese kot mera podobnosti. Izračunamo jo po sledeči enačbi:

$$r(u_i, u_j) = \frac{\sum_{k=1}^{n} (e(u_i, h_k) - \bar{e}(u_i))(e(u_j, h_k) - \bar{e}(u_j))}{\sqrt{\sum_{k=1}^{n} (e(u_i, h_k) - \bar{e}(u_i))^2 \sum_{k=1}^{n} (e(u_j, h_k) - \bar{e}(u_j))^2}} \tag{4.6}$$

$r(u_i, u_j)$ predstavlja korelacijo med aktivnim uporabnikom u_i ter potencialnim sosedom u_j, izračunana glede na vse vsebine h_k, ki sta jih ocenila oba uporabnika. Ocena za izbrano vsebino je predstavljena z $e(u_i, h_k)$ za aktivnega uporabnika in $e(u_j, h_k)$ za potencialnega soseda. Vrednosti $\bar{e}(u_i)$ ter $\bar{e}(u_j)$ pa predstavljata povprečne ocene obeh uporabnikov.

Tanimotova mera podobnosti

Tanimotova mera podobnosti je enaka kosinusu kote med dvema vektorjema ocen. Vsak krajevni vektor nam predstavlja nabor ocen, ki jih je uporabnik zabeležil v sistem. Izračunane podobnosti zavzemajo vrednosti med 0 in 1. Če sta si vektorja popolnoma enaka, je mera podobnosti enaka 1, če sta si popolnoma različna, pa 0. Izračunamo jo po sledeči enačbi:

$$sim(u_a, u_j) = \frac{u_a^T u_j}{\|u_a\| \|u_j\|} \tag{4.7}$$

kjer $sim(u_a, u_j)$ predstavlja izračunano mero podobnosti, u_a profil altovnega uporabnika, kateremu iščemo sosede, u_j pa profil potencialnega soseda.

Fujeva mera podobnosti

Fujeva mera podobnosti predstavlja razširjeno verzijo Tanomotove mere podobnosti, ker je sposobna zaznati tudi, če sta si dva vzorca negativno linearno odvisna. V tem primeru je vrednost mere podobnosti enaka -1. Izračunamo jo po sledeči enačbi:

$$sim(u_a, u_j) = \frac{D_E(u_a, u_j)}{\|u_a\| + \|u_j\|} \tag{4.8}$$

kjer $sim(u_a, u_j)$ predstavlja izračunano mero podobnosti, u_a profil aktivnega uporabnika, kateremu iščemo sosede, u_j pa profil potencialnega soseda. $D_E(u_a, u_j)$ pa predstavlja izračunano Evklidovo razdaljo med obema profiloma po enačbi 4.1.

Izbira uporabljenih metod

Prvi kriterij, po katerem lahko ločimo razdalje med seboj, je pomen vrednosti, ki nam jih vračajo. Prve (Evklidova, Manhattan, Čebiševa, Mahalanobiseva razdalja ter utežena razdalja Minkovskega) nam vračajo številske vrednosti, ki nam predstavljajo absolutno razdaljo med primerjanimi vzorci – uporabniki. Pearsonov korelacijski koeficient, Tanomotova mera podobnosti in Fujeva mera podobnosti, pa nam vrnejo številsko vrednost, ki predstavlja, v kakšnem odnosu sta si vzorca (torej relativno mero razdalje) – uporabnika med seboj – kako dobro so ocene obeh povezane z regresijsko premico. Pri Uporabniškem modeliranju nas odnos med uporaniki zelo zanima, zato so te tri mere boljši kandidati za naš sistem. Ker smo želeli začeti z računsko manj zahtevnimi postopki (tudi za manjšo računsko zahtevnost končnega sistema, smo zato izbrali Pearsonov korelacijski koeficent kot prvo metodo, s katero smo zasnovali naš sistem.

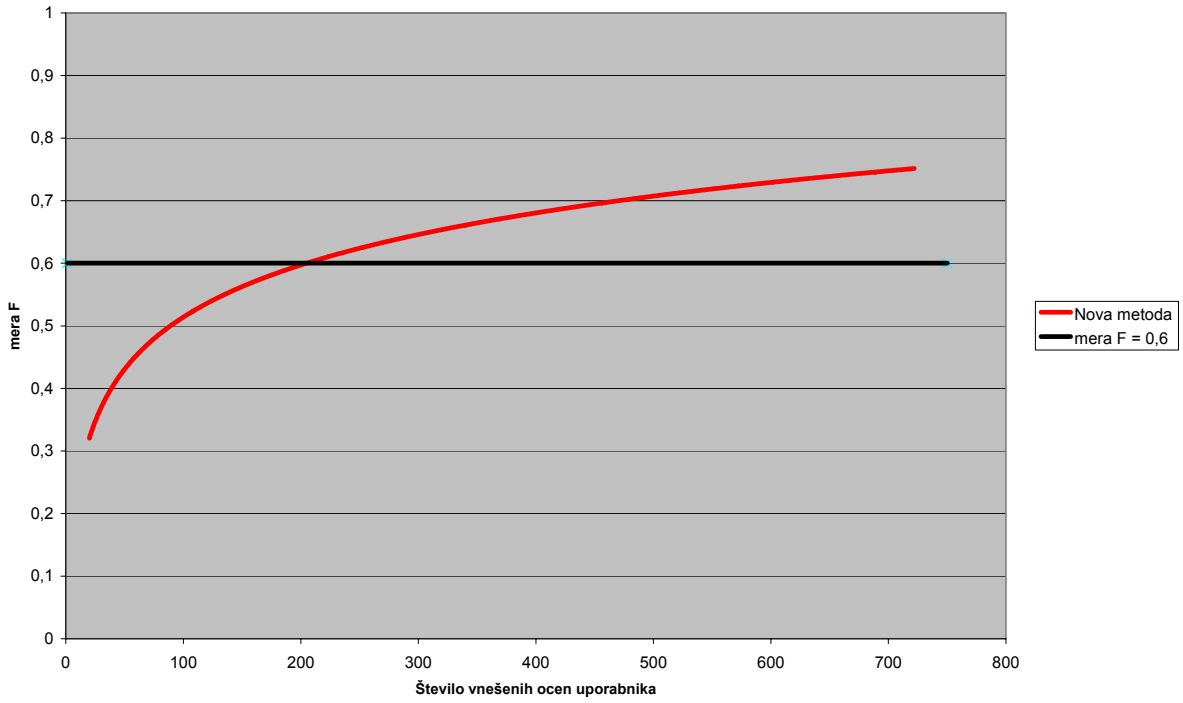
Tipične vrednosti podobnosti, izračunane s Pearsonovim korelacijskim koeficientom, potekajo od -1 (negativna linearna odvisnost) do +1(pozitivna linearna odvisnost). Vrednost +1 nam kaže na to, da bi bil uporabnik zelo primeren za najbližjega soseda, vrednost 0 ali -1 pa da je popolnoma neprimeren za to vlogo. Pri izračunavanju koeficienta po enčbi 4.9 [8] smo naleteli na težavo pri deljenju z zelo majhnimi števili v programskem okolju Java, zato smo izračun ustrezno priredili. Če smo opazili, da pride do deljenja z zelo majhnim številom, to pomeni, da uporabnik nikakor ni primeren za najbližjega soseda in smo zato vrednost podobnosti avtomatsko nastavili na 0. Na ta način preprečimo, da bi prišli v situacijo, kjer bi vrednost podobnosti privzela neprimerne vrednosti. Predvidevamo, da imamo v sistemu dovolj sosedov, da bomo vedno lahko našli dovolj uporabnikov, primernih za najbližje sosede. Potrebno je tudi omeniti, da uspešnosti metode za iskanje najbližjih sosedov ni možno direktno izmeriti. Končni rezultat iskanja najbližjih sosedov je vedno spisek **N** sosedov in samo s pomočjo tega spiska ne moremo ugotoviti ali so izbrani sosedje primerni ali ne. Primernost sosedov lahko ugotovimo, šele na podlagi njihovih profilov, ko uporabniku poiščemo primerne vsebine, jih ponudimo in dobimo povratno informacijo (oceno) o ponujenih vsebinah. Zato smo v okviru našega razvoja metode primernost opisanih pristopov merili tako, da smo rezultate prve faze uporabili v drugi fazi. Za vse profile oz. izbrane sosede je druga faza delovala enako in so bili končni rezultati odvisni samo od primernosti izbranih sosedov. Za mero uspešnosti smo uporabili mere natančnosti, deleža pravilno najdenih vsebin ter mero F, ki so bile podrobneje opisane v poglavju 3.4.1.

4.3.1 Pearsonov korelacijski koeficient

Pearsonov korelacijski koeficient [8] računamo po formuli 4.6. Za potrebe računanja podobnosti med posameznimi uporabniki nato definiramo mero podobnosti $sim(u_a, u_j)$, ki je enaka vrednosti korelaciji, kot je razvidno iz enačbe 4.10.

$$sim(u_i, u_j) = r(u_i, u_j) \tag{4.9}$$

Pred računanjem podobnosti je bilo potrebno v vsak uporabniški profil vnesti povprečno oceno uporabnika. Med iskanjem primernih sosedov sistem nato za vsakega potencialnega soseda izračuna vrednost podobnosti ter jo zabeleži v temu namenjeni tabeli (tabela je zelo enostavna, saj vsebuje samo imena uporabnikov ter vrednosti podobnosti med njimi). Na koncu postopka sistem iz tabele izbere 50 najbolj primernih sosedov ter se s pomočjo njihovih profilov poišče primerne vsebine. Rezultati iskanja najbližjih sosedov s pomočjo te metode so razvidni na sliki 18.



Slika 18: Rezultati prilagojene utežene vsote ter Pearsonovega korelacijskega koeficienta

Iz slike 18 je razvidno, da je metoda primerna in prične hitro delovati z veliko gotovostjo (vrednost mere F nad 0.6). Za doseženo mejo zadostuje že manj kot 200 vnesenih ocen vsebine s strani uporabnika. Za sistem, ki je mišljen za vsakodnevno uporabo, to število ne predstavlja velikega problema. Ker sklepamo, da bi povprečni uporabnik skoraj vsak dan v sistem vnesel kako oceno, bi do številke 200 prišli zelo hitro. Ob pregledu posameznih rezultatov pa vseeno opazimo, da se tudi pri večjem številu vnesenih ocen še vedno pojavljajo primeri, ko sistem odpove in uporabniku ne more predlagati vsebin z dovolj veliko mero gotovosti. Po podrobnejši preučitvi takšnih primerov smo ugotovili, da prihaja do dveh možnih scenarijev:

- Uporabnik vsebin ni ocenjeval ocen skladno s svojim okusom (se pravi ni odkrito podajal ocen temveč jih je prilagajal) oziroma njegov okus tako izstopa. Zato ni možno odkriti uporabnikov s podobnim okusom, ker so prekrivanja med aktivnim uporabnikom ter potencialnimi sosedi tako majhna, da je vsak uporabnik enako dober potencialni sosed.
- Uporabnik je vsebine ocenjeval skladno s svojim okusom in je v večini primerov ocenil samo tiste vsebine, ki so mu res všeč, in jih ocenil z bolj ali manj enako oceno. Do take situacije lahko pride tudi, kadar se uporabnik prvič prijavlja v sistem in želi sistemu pomagati tako, da takoj označi vse vsebine, ki so mu res najbolj všeč.

V prvem primeru sistem rezultatov ne more izboljšati, ker problem ni v izbranem pristopu temveč v samih uporabnikovih preferencah. V drugem primeru bi moral sistem v resnici najbolje delovati, saj je uporabnik jasno povedal kaj mu je všeč in kaj ne. Podrobnejši pregled ocen uporabnikov v teh situacijah je pokazal, da uporabniki v teh primerih vse vsebine ocenjujejo z oceno, ki zelo malo odstopa od njihove povprečne ocene. Pri izračunu mere podobnosti pride zato do situacije, v kateri prihaja do numeričnih napak zaradi računanja z zelo majhnimi števili. Posledično se vsi potencialni sosedje obravnavajo enako oz. je prispevek njihovih ocen prezrt. Numerične napake lahko odpravimo s preureditvijo enačbe v:

$$r(u_i,u_j)=\sum_{k=1}^{n}\frac{(e(u_i,h_k)-\bar{e}(u_i))}{\sigma_{u_i}}\frac{(a(u_j,h_k)-\bar{e}(u_j))}{\sigma_{u_j}} \tag{4.10}$$

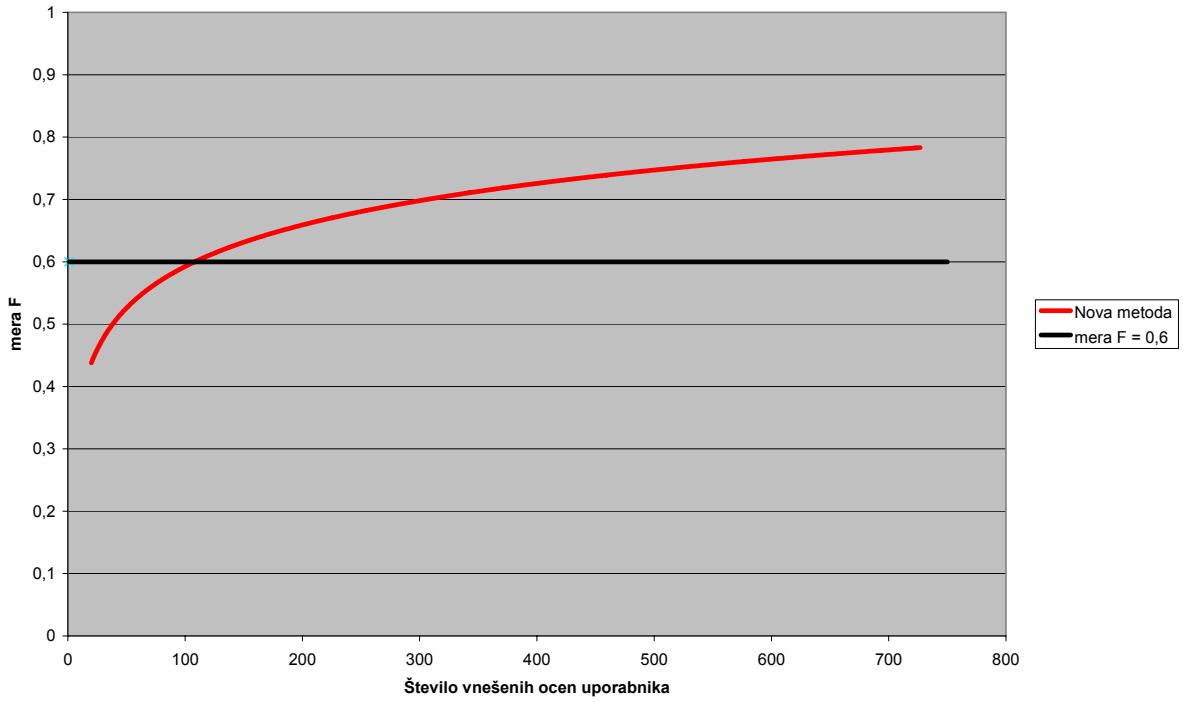
Kjer nam $r(u_i,u_j)$ predstavlja korelacijo med aktivnim uporabnikom u_i ter potencialnim sosedom u_j, izračunana glede na vse vsebine h_k, ki sta jih ocenila oba uporabnika. Ocena za izbrano vsebino je predstavljena z $e(u_i,h_k)$ za aktivnega uporabnika in $e(u_j,h_k)$ za potencialnega soseda. Vrednosti $\bar{e}(u_i)$ ter $\bar{e}(u_j)$ pa predstavljata povprečne ocene obeh uporabnikov. Vrednosti σ_{u_i} ter σ_{u_j} pa predstavljata standardni odklon uporabnikovih ocen.

Ker vsi sistemi ne omogočajo naprednejših izračunov in ker se je spremenjena formula izkazala za bolj kompleksno, smo izbrali drugačno rešitev – vpeljavo alternativnega iskanja podobnih uporabnikov. Ker je bil eden od virov težav deljenje z zelo majhnimi številkami pri izračunu podobnosti, smo se odločili uporabiti enostavnejši pristop, oziroma bolj preprosto enačbo. Na osnovi informacij v literaturi smo ugotovili, da se na drugih področjih pogosto uporablja kot mera razdalje med dvema točkama tudi Evklidova razdalja [2, 39]. Zato smo želeli ugotoviti, če ta metoda deluje tudi pri računanju podobnosti med uporabniki.

4.3.2 Evklidova razdalja

Evklidsko razadljo smo izbrali izmed p-norm kot uravnotežen primer med skrajnostima $p=1$ (Manhattan razdalja) ter $p=\infty$ (Čebiševa razdalja) norma. $p=1$ norma v večji meri izbere veliko število manjših odklonov, max norma $p=\infty$ pa po drugi strani upošteva vsak velik odklon. Evklidsko razdaljo [42] računamo po sledeči enačbi 4.1.

Tako kot pri prejšnji metodi smo izračunane vrednosti podobnosti shranili v posebno tabelo in na koncu postopka iz nje izbrali 50 najbolj primernih sosedov. Rezultati iskanja najbližjih sosedov s pomočjo te metode so razvidni na sliki 19:



Slika 19: Rezultati prilagojene utežene vsote ter Evklidove razdalje

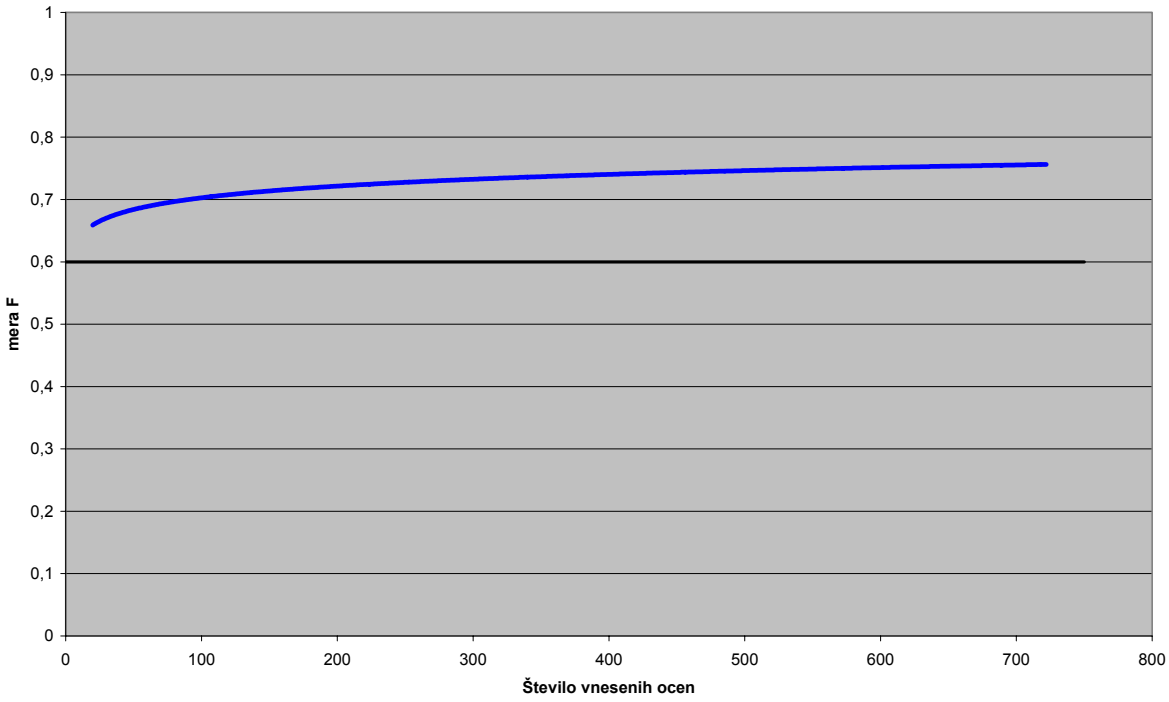
Rezultati pokažejo, da metoda deluje s podobno mero gotovosti kot Personov koeficient oz. celo bolje, saj hitreje doseže visoko stopnjo gotovosti (mera F nad 0.6 – glej 3.4). Vrednost 0.6 preseže ponekod že pri samo 100 vnesenih ocenah.

4.3.3 Kombinacija obeh metod

Rezultati, prikazani v prejšnjih poglavijih, so pokazali, da sta obe metodi primerni za iskanje najbližjih sosedov. Podrobnejša analiza je pokazala, da Evklidska razdalja zanesljivo deluje v vseh primerih, saj je njen izračun neodvisen od uporabnikove povprečne ocene (torej se tako izognemo numeričnim napakam, o katerih smo pisali v prejšnjem poglavju). Izkazalo se je tudi, da kadar obe metodi vrneta smiseln rezultat, Pearsonov koeficient deluje bolje. Zato smo se odločili v končnem sistemu uporabiti obe metodi in med njima preklapljati, odvisno od lastnosti aktivnega uporabnika, glede na odstopanje uporabnika od lastne povprečne ocene. Vzporedno uporabo obeh metod hkrati omogoča tudi dejstvo, da obe metodi vračata pri izračunu podobnosti številske vrednosti kot mero podobnosti. Zato ni potrebno dodajati modulov, ki bi izračunane podobnosti pretvarjali in usklajevali za kasnejšo rabo pri iskanju primernih vsebin.

Potrebno je bilo ugotaviti, kdaj uporabiti katero od metod. Ker je naš uporabniški model enostaven in ne vsebuje veliko podatkov za analizo, smo se odločili da ohranimo želeno enostavnost sistema in zato odločitev osnujemo na podlagi odstopanja od popvrečne ocene (glej 4.3.1). Kot smo ugotovili pri preskušanju Pearsonovega koeficienta, je ključnega pomena uporabnikovo odstopanje od lastne povprečne ocene. Zato smo se odločili, da bo to odstopanje služilo kot odločitev za izbiro uporabljene metode za iskanje sosedov. Kadar uporabnik močno odstopa od lastne povprečne ocene (to se ponavadi zgodi na začetku ko se sistem šele bolj podrobno uči o uporabnikovih preferencah), bo sistem uporabil Pearsonov koeficient, če pa odstopanje pade pod izbrano mejo, preklopi na Evklidsko distanco. Na sliki

16 je prikazan potek sistema, pri katerem smo za predikcijo uporabili metodo prilagojene utežene vsote (glej 4.4.3), med metodami za iskanje primernih vsebin pa smo preklaplali glede na to ali je uporabnikov standardni odklon presegal mejo 0,1 ali ne. (psevdokoda postopka ja zapisana v dodatku A.4) Vidimo, da je učinkovitost sistema glede na F mero opazno boljša kot učinkovitost posameznih metod, ki smo jih opisali v prejšnjih podpoglavjih.



Slika 20: Preklop distanc

S kombiniranjem oziroma preklaplanjem med obema metodama (glej psevdokodo v dodatku A.4) smo tako dobljene rezultate sistema izboljšali. Ker tako dobljeni sistem uporabniku primerne vsebine išče z veliko zanesljivostjo (torej z veliko vrednostjo mere F), smo se odločili, da bi bilo dodajanje še več metod za iskanje sosedov v sistem nepotrebno, zato smo se raje osredotočili na metode iskanja primernih vsebin.

4.4 Izračun napovedane ocene

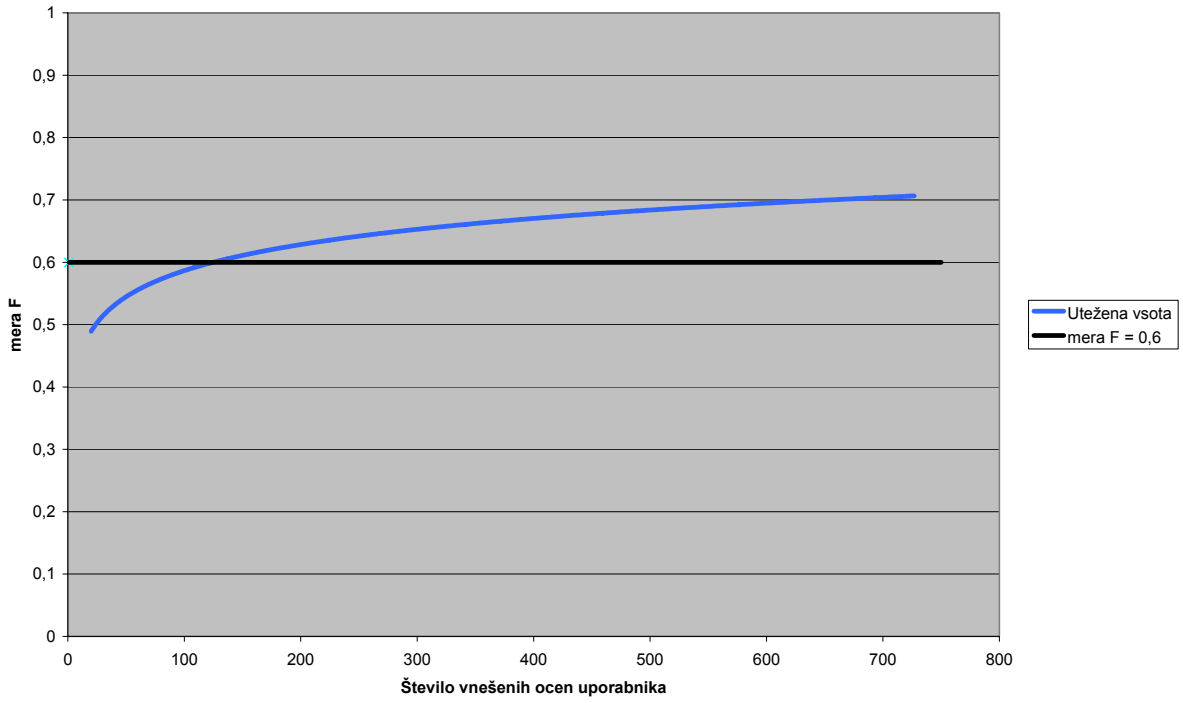
Ko sistem zgradi uporabnikov profil $up(u_a)$ in vanj zabeleži najbližje sosede u_s, preide v drugo fazo in začne z iskanjem uporabniku primernih vsebin. Primernost vsebine lahko izrazimo na več načinov. Pogosto primernost vsebine predstavimo z napovedano oceno $\hat{e}(u_a, h_p)$, ki predstavlja napoved sistema, kako bo uporabnik vsebino dejansko ocenil. Pri metodah, ki vračajo primernost v tej obliki, je merjenje uspešnosti enostavno, saj lahko direktno primerjamo napovedano oceno $\hat{e}(u_a, h_p)$ z dejansko uporabnikovo oceno $e(u_a, h_p)$. Izmed obstoječih metod smo izbrali dva pristopa: – **uteženo vsoto** ter **Bayesov približek**. Naknadno pa smo vpeljali še novo metodo – **prilagojeno uteženo vsoto**, ki smo jo razvili na podlagi rezultatov preizkušanja prvih dveh metod.

4.4.1 Utežena vsota

Utežena vsota [36] uporablja povprečno oceno aktivnega uporabnika in ocene njegovih sosedov. Izračunamo jo po sledeči enačbi:

$$\hat{e}(u_a, h_k) = \bar{e}(u_a) + \sum_{j=1}^{n} sim(u_a, u_j)(e(u_j, h_k) - \bar{e}(u_j)) \tag{4.11}$$

V enačbi je $\hat{e}(u_a, h_k)$ izračunana napovedana ocena za vsebino h_k za aktivnega uporabnika u_a. Vrednost $\bar{e}(u_a)$ je povprečna ocena aktivnega uporabnika, $sim(u_a, u_j)$ izračunana podobnost med aktivnim uporabnikom in njegovim sosedom j. Ocena soseda j za vsebino h_k je označena z $e(u_j, h_k)$, povprečna ocena soseda j pa z $\bar{e}(u_j)$. Napovedana ocena za izbrano vsebino je izračunana na podlagi povprečne ocene aktivnega uporabnika in prispevkov vseh sosedov, ki so to vsebino že ocenili. Pri tem je potrebno opozoriti na to, da ta metoda utežene vsote upošteva samo prispevke sosedov, torej mnenje vseh ostalih uporabnikov, ki so izbrano vsebino tudi ocenili, pri tej metodi nima vpliva. Rezultati testiranja metode so razvidni iz slike 21.



Slika 21: Potek mere F pri uteženi vsoti

Iz grafa vidimo, da metoda zelo hitro doseže mejo mera F = 0.6, vendar z večanjem števila glasov zelo počasi raste. Sistem bo torej v vseh fazah enako zanesljivo deloval. Podrobneje bomo metode ocenili v poglavju 4.5.1, kjer bomo primerjali vse preizkušene metode.

Omejitev metode utežene vsote je, da išče potencialno zanimive vsebine samo na podlagi ocen, podanih s strani sosedov. Torej lahko zgreši potencialno zanimive vsebine, če jih nihče izmed sosedov še ni ocenil. Vprašljiva je tudi zanesljivost izračunane napovedane ocene, kadar je le-ta zasnovana na podlagi ocen, ki jih je podal samo majhen delež vseh izbranih sosedov – torej ker sta vsebino ocenila na primer samo dva izmed petdestih sosedov. V takih

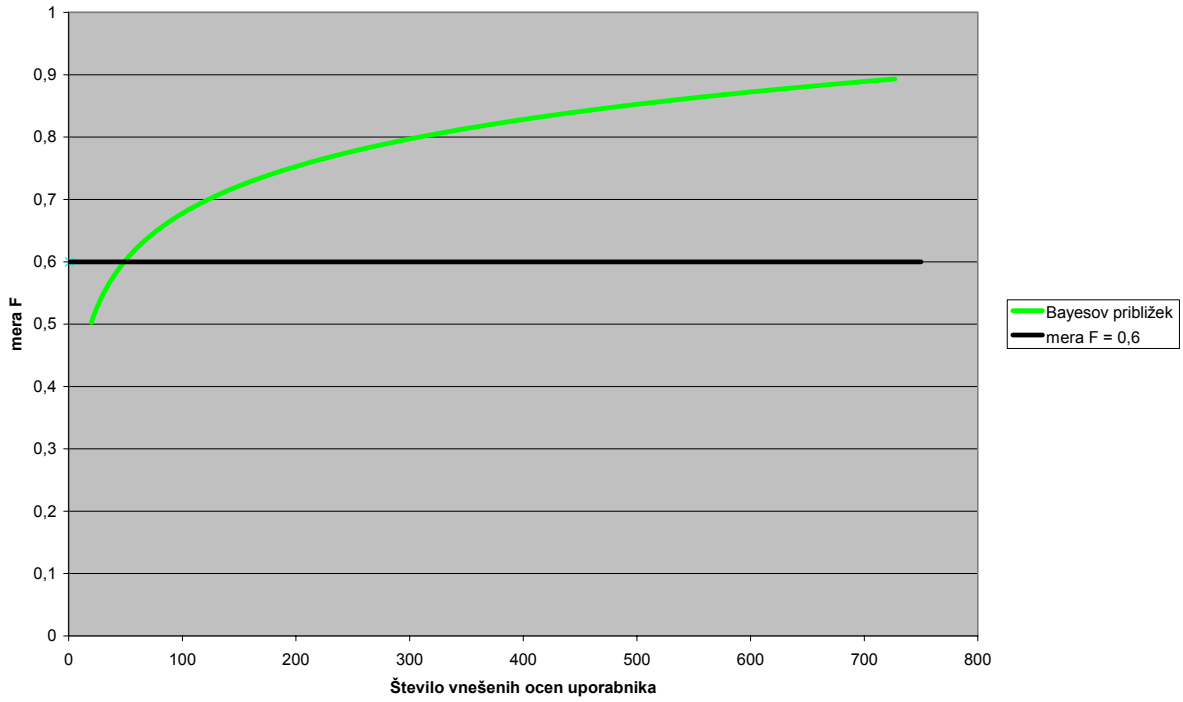
primerih se metoda zato lahko zelo slabo obnese. Rešitev je lahko predelava obstoječe metode tako, da daje prednost vsebinam, ki jih je ocenilo čimveč sosedov ali pa vpeljava nove metode, ki poleg najbližjih sosedov upošteva tudi druge uporabnike. V ta namen smo v sistem dodali metodo Bayesovega približka, ki jo podrobneje opisujemo v sledečem poglavju.

4.4.2 Bayesov približek

Želeli smo oceniti, s kakšno gotovostjo bo deloval sistem, ki bo poleg mnenja najbližjih sosedov upošteval tudi mnenje celotne publike (vseh uporabnikov sistema). Tak sistem bi lahko zaznal tudi vsebine, ki so širši publiki zelo všeč, sosedje pa jih še niso ocenili. Z uporabo metode utežene vsote takih vsebin nismo sposobni zaznati. Vpliv celotne publike pa ima tudi negativne posledice, saj sistem včasih ne izračuna pravilno napovedane ocene za uporabnika z okusom, ki opazno odstopa od okusa celotne publike. Za pristop, ki izpolnjuje te zahteve, se najbolj pogosto uporablja metodo Bayesovega približka [59], ki se izračuna po sledeči enačbi:

$$\hat{e}(u_a, h_k) = \frac{n}{n+m}\bar{e}_s(h_k) + \frac{m}{n+m}\bar{e}_p(h_k) \tag{4.12}$$

V enačbi je $\hat{e}(u_a, h_k)$ izračunana napovedana oceno za vsebino h_k za aktivnega uporabnika u_a. V izračunu sta uporabljeni dve povprečni oceni: $\bar{e}_s(h_k)$ je vrednost povprečne ocene za vsebino h_k vseh sosedov, ki so to vsebino ocenili, $\bar{e}_p(h_k)$ pa predstavlja vrednost povprečne ocene za vsebino h_k izračunane na podlagi ocen vseh uporabnikov v sistemu, ki so to vsebino ocenili. Parameter n predstavlja število sosedov, ki so vsebino ocenili, m pa predstavljata utež, s katero izbiramo, kolikšen pomen pripišemo mnenju najbližjih sosedov oz. vseh uporabnikov sistema (izbira vrednosti uteži je podrobneje opisana v poglavju 5.2). S pomočjo teh uteži sistemu določimo, ali daje prednost ocenam najbližjih sosedov ali pa ocenam vseh obstoječih uporabnikov. Prednost Bayesovega približka je, da upošteva tudi splošno mnenje vseh uporabnikov o vsebini, namesto samo ocen najbližjih sosedov. Na ta način lahko za določeno vsebino napovemo oceno tudi, če jo je ocenilo zelo malo sosedov. Rezultati testiranja metode so razvidni iz slike 22:



Slika 22:Potek mere F pri Bayesu

Iz rezultatov vidimo, da metoda zelo hitro preseže mejo mera F = 0.6 in da se z naraščanjem števila vnesenih glasov tudi vztrajno veča stopnja zanesljivosti, s katero metoda išče primerne vsebine za uporabnika. Rezultate podrobneje komentiramo v poglavju 4.5.1, kjer metode primerjamo med seboj.

4.4.3 Prilagojena utežena vsota

Po pregledu uporabljenih metod in dobljenih rezultatov smo ugotovili, da ima pri metodi utežene vsote (glej 4.4.1), povprečna ocena aktivnega uporabnika zelo veliko vlogo pri izračunu napovedane ocene

Ugotovili smo, da lahko metodo utežene vsote izboljšamo. Kot je opisano v enačbi v poglavju 4.4.1, ima vlogo pri izračunu napovedane ocene tudi povprečna ocena aktivnega uporabnika. Pri večini uporabnikov to ni problematično, vendar smo odkrili dve možnosti, pri katerih lahko pride do težav:

- Kadar je uporabnik vsebine ocenjeval zelo pozitivno, je njegova povprečna ocena zelo visoka (na primer 8.5 na lestvici od 0 do 10). Posledica tega je, da sistem za vse vsebine izračuna visoko napovedano oceno, kljub temu da so sosedje vsebino negativno ocenili.
- Druga težava, na katero naletimo, je ravno obratna: če je uporabnik veliko vsebin negativno ocenil, je njegova povprečna ocena zelo nizka (recimo 1.5 na lestvici od 0 do 10). V tem primeru se lahko zgodi, da sistem za vse vsebine izračuna nizko napovedano oceno kljub pozitivnim ocenam s strani najbližjih sosedov.

V obeh primerih sistem ne deluje pravilno, ker na koncu predlaga vse potencialne vsebine ali pa nobene. Do težav pride, ker v primeru ekstremno visoke ali nizke povprečne ocene aktivnega uporabnika prispevki s strani najbližjih sosedov niso dovolj visoki, da bi omogočili

pravilni izračun napovedane ocene. Če pa vsebine niso ocenili vsi sosedje, je teh prispevkov še manj in zato še težje spremenijo napovedno oceno.

Na osnovi teh ugotovitev smo se odločili uteženo vsoto prirediti tako, da pri izračunu povprečne ocene aktivnega uporabnika ne upošteva več. Napovedano oceno izračunamo po sledeči enačbi:

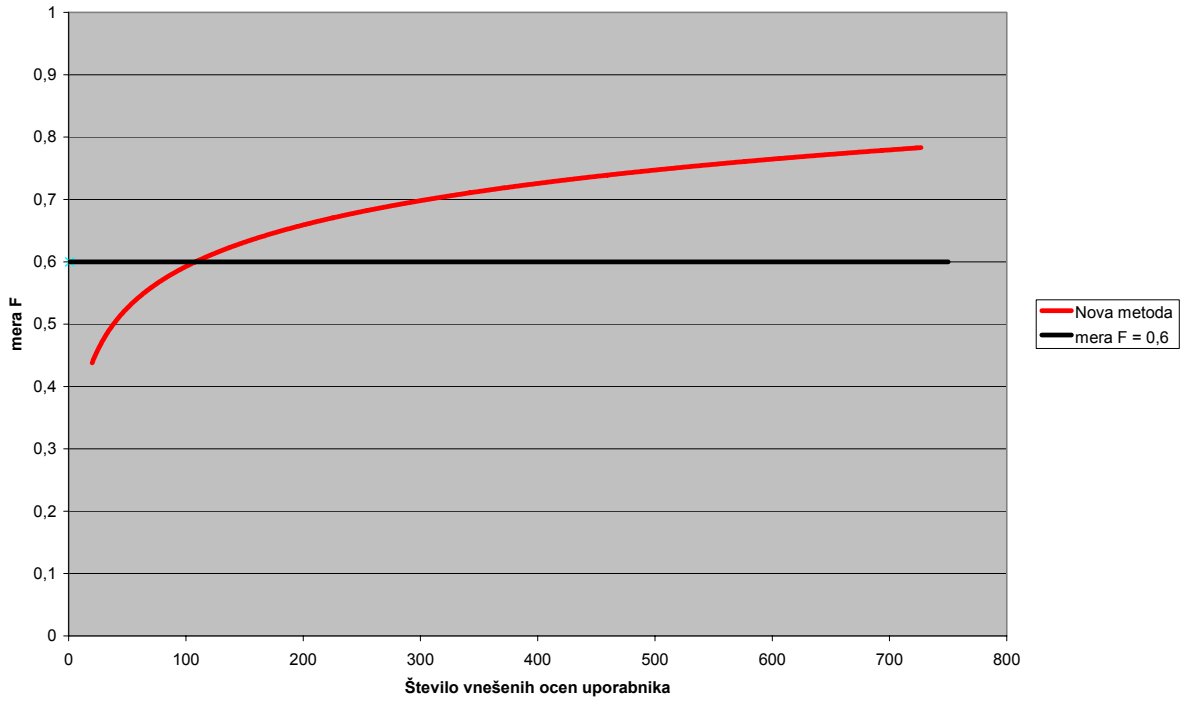
$$\hat{e}(u_a, h_k) = \sum_{k=1}^{n} sim(u_a, u_j)(e(u_j, h_k) - \bar{e}(u_j)) \tag{4.13}$$

V enačbi je $\hat{e}(u_a, h_k)$ izračunana napovedana ocena za vsebino h_k za aktivnega uporabnika u_a. Vrednost $sim(u_a, u_j)$ je izračunana podobnost med aktivnim uporabnikom in njegovim sosedom j. Ocena soseda j za vsebino h_k je predstavljena z $e(u_j, h_k)$, povprečna ocena soseda j pa z $\bar{e}(u_j)$. Posledica te spremembe je, da sedaj damo večji pomen mnenju sosedov o potencialni vsebini, hkrati pa rezultat metode ni več dejanska ocena, temveč relativna mera. Sistem torej ni več zmožen direktno napovedati, kako bo uporabnik ponujeno vsebino dejansko ocenil. Izračunana vrednost predstavlja, kako so se sosedje odzvali na vsebino. Pozitivna vrednost pomeni, da je večina sosedov vsebino ocenila z oceno, višjo od svoje povprečne ocene, kar pomeni, da so se na vsebino pozitivno odzvali. Negativna izračunana vrednost pa pomeni, da so ocenili z oceno, nižjo od svoje povprečne, in to interpretiramo kot negativno reakcijo na ponujeno vsebino. Za predlagane vsebine izberemo tiste z najvišjo izračunano vrednostjo. Za testiranje nove metode je bilo potrebno spremeniti vrednotenje uspešnosti sistema – izračun vrednosti natančnosti (P), deleža pravilno najdenih vsebin (R) ter mere F. Ker metoda ne vrača več napovedane ocene, smo morali prirediti določanje vrednosti Pravilno izbranih, Napačno izbranih, Pravlino zavrnjenih ter Napačno zavrnjenih vsebin (glej poglavje 3.4.1). Postopke smo prilagodili na sledeč način:

- Za **Pravilno izbrane** vsebino označimo, če je bila označena za pozitivno sprejeto s strani sosedov in je bila uporabnikova dejanska ocena nad zadano mejo.
- Za **Napačno izbrane** vsebino označimo, če je bila označena za pozitivno sprejeto s strani sosedov, vendar je bila uporabnikova dejanska ocena pod zadano mejo.
- Za **Pravilno zavrnjene** vsebino označimo, če je bila označena za negativno sprejeto s strani sosedov in je bila tudi uporabnikova dejanska ocena pod zadano mejo.
- Za **Napačno zavrnnjene** pa vsebino označimo, če je bila označena za negativno sprejeto s strani sosedov, vendar je bila uporabnikova dejanska ocena nad zadano mejo.

Za mejo pozitivnih ocen (torej za zadano mejo) smo ohranili vrednost 0.7 (glej poglavje 3.4.1)

Na osnovi teh prilagoditev smo nato lahko izračunali vrednosti natančnosti, deleže pravilno najdenih vsebin ter mere F. Rezultati testiranja nove metode so prikazani na sledeči sliki:



Slika 23: Potek mere F pri prilagojeni uteženi vsoti

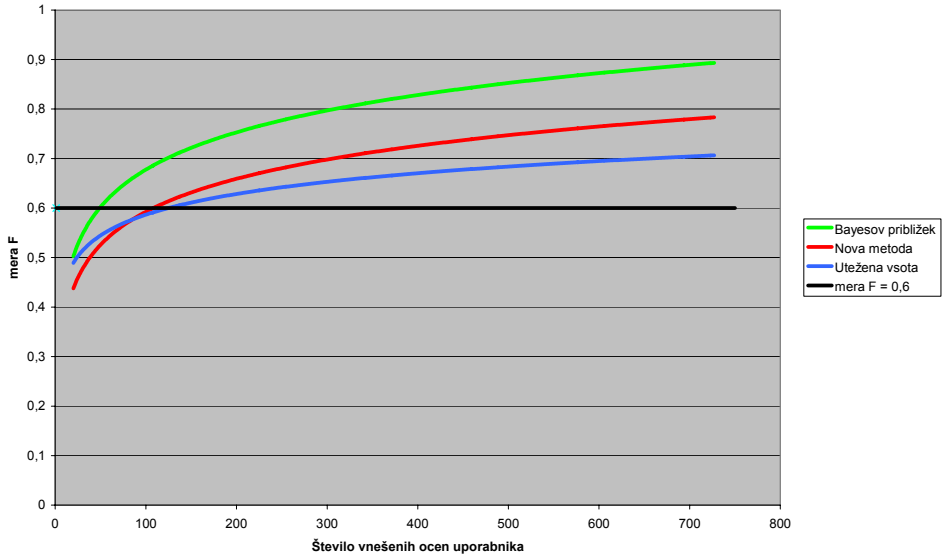
Iz rezultatov vidimo, da nam je uspelo izboljšati metodo utežene vsote in se približati rezultatom, dobljenim z Bayesovim približkom. Slabost prilagojene vsote, razvidna iz slike 23, je, da metoda v zagonu potrebuje več glasov, da začne delovati z visoko gotovostjo (visoka vrednost mere F), čeprav jo še vedno doseže v prej kot 150 vnesenih ocenah. Metodo lahko uspešno uporabimo namesto utežene vsote.

4.5 Rezultati

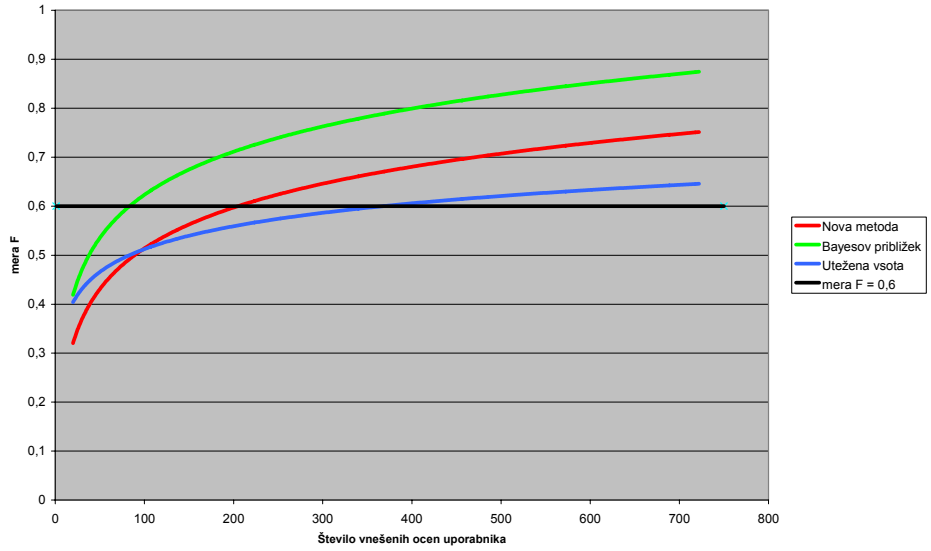
Pri vrednotenju učinkovitosti sistema smo se osredotočili na dve področji – uspešnost metode pri iskanju najbližjih sosedov in zanesljivost metode pri iskanju primernih vsebin. Metode iskanja najbližjih sosedov smo ovrednotili že v poglavju 4.3 ter ugotovili, da je najbolj primerna kombinacija obeh.

4.5.1 Primerjava metod za iskanje primernih vsebin

Potem, ko smo sistem testirali z vsako izmed treh izbranih metod iskanja primernih vsebin, smo imeli na voljo rezultate, s pomočjo katerih lahko metode primerjamo med seboj in izberemo najbolj primerno. Rezultati primerjave so razvidni iz slik 24 in 25.



Slika 24: Primerjava metod – Evklidova razdalja



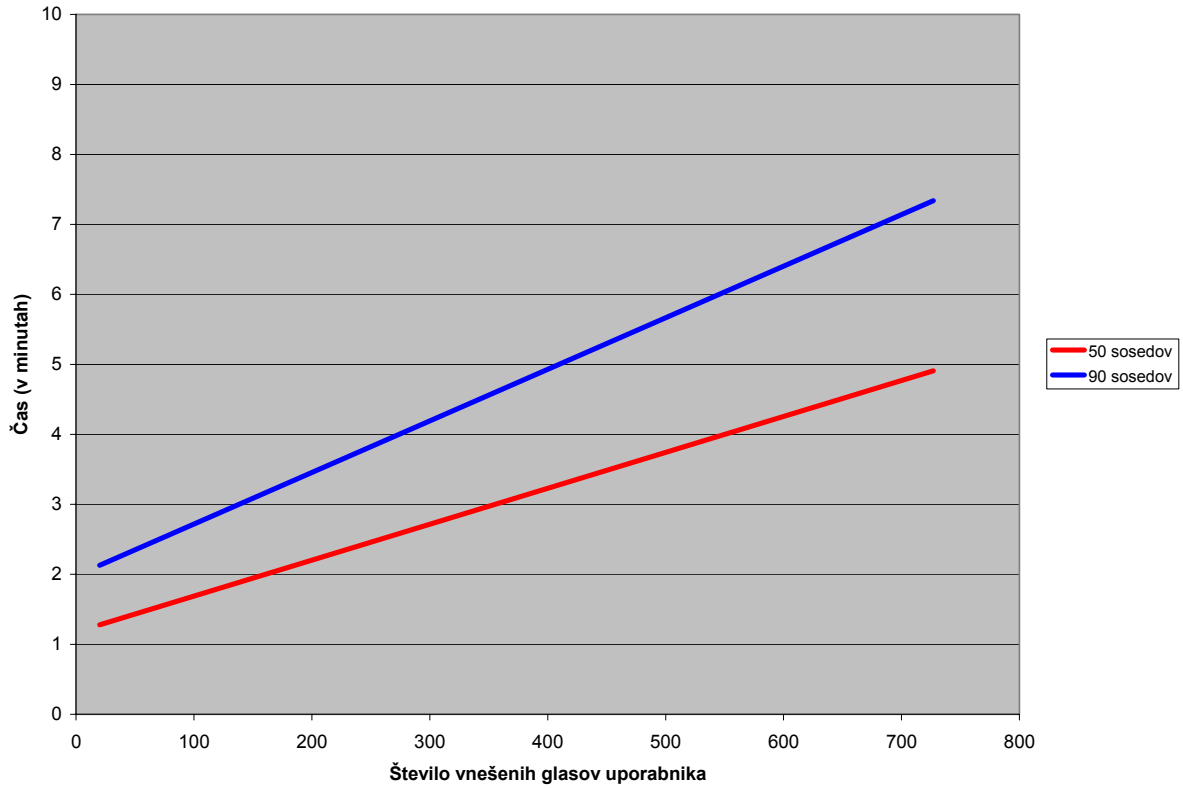
Slika 25: Primerjava metod – Pearsonov korelacijski koeficient

Rezultati pokažejo, da se najbolje obnese metoda Bayesovega približka, sledi pa ji naša metoda prilagojene uteženе vsote. Vse tri metode imajo dobre rezultate, saj pri zabeleženih 70 ocenah uporabniku pravilno predlagajo že več kot polovico vsebin. Statistična primerjava metod (glej dodatek B.1) pokaže, da na začetku (manj kot 100 vnesenih glasov) vse tri metode delujejo s podobno zanesljivostjo. Ko je uporabnik vnesel več kot 100 glasov pa se Bayesova metoda značilno bolje obnese (za uporabnike z več kot 300 vnesenimi glasovi statistična primerjava ni več zanesljiva, ker imamo premalo uporabnikov).

Za dokončno izbiro metode iskanja primernih vsebin je potrebno poznati tudi okolje, v katerem bo sistem deloval. Če bo sistem deloval v okolju z veliko količino aktivnih uporabnikov, ki pogosto vnašajo ocene, je najbolj primerna izbira gotovo Bayesov približek, saj le-ta upošteva tudi mnenje celotne publike. V aktivnem okolju je mnenje publike ažurno in se zato nanj splača zanašati. Če pa računamo, da bo sistem deloval v okolju, kjer se uporabniki le redko odzovejo na nove vsebine, so primernejše druge metode.

4.5.2 Čas procesiranja ocen uporabnikov

Čeprav je sistem deloval zelo učinkovito, z dobrimi rezultati iskanja primernih vsebin, se je izkazalo, da je časovno zelo zahteven. Za izgradnjo modela posameznega uporabnika in iskanje primernih vsebin zanj je porabil veliko časa. Podrobnejša analiza časovnega poteka pri izgradnji uporabniškega modela je prikazana na sliki 26.



Slika 26: Časovna zahtevnost v odvisnosti od števila vnesenih ocen uporabnika

Na sliki sta prikazana dva poteka časovne zahtevnosti (predstavljena z interpoliranima premicama, osnovanima na 612665 točkah). Z rdečo je prikazan čas, potreben za iskanje primernih vsebin uporabniku v sistemu, ki uporabi 50 najbližjih sosedov, z modro pa sistem z

90 najbližjimi sosedi. Za uporabnika s povprečnim številom vnesenih glasov naš sistem porabi do 6 minut časa, kar ni primerno za uporabo v realnih sistemih.

S pomočjo analize uporabljenih postopkov in algoritmov smo računsko zahtevnost sistema ocenili na $m^2 \cdot n + O(mn)$ (glejte dodatek A), kjer m predstavlja število vseh uporabnikov v sistemu, n pa število vseh vsebin, ki so uporabnikom na voljo. Pri tem smo za osnovno operacijo privzeli cikel Intelovega procesorja.Ker obe števili med uporabo sistema naraščata (dodajo se nove vsebine, registrirajo se novi uporabniki), skladno raste tudi število računskih operacij, ki jih potrebujemo da posameznemu uporabniku najdemo primerne vsebine.

4.6 Možnosti za nadaljnji razvoj

Postavili in preizkusili smo sistem za uporabniku prilagojeno iskanje vsebin. Uspešno smo preizkusili dve metodi za iskanje najbližjih sosedov, dve metodi za iskanje uporabniku zanimivih vsebin in na podlagi rezultatov razvili povsem novo metodo za iskanje vsebin. Uspešno smo uvedli inovativen pristop za iskanje najbližjih sosedov, ki povezuje dve metodi hkrati. Rezultati sistema so zelo dobri, saj sistem z zelo majhnim številom vnesenih ocen dosega visoko stopnjo gotovosti ter tako uporabnikom že zgodaj ponuja zanimive vsebine.

Sistem je še v razvojni fazi in naleteli smo na določene težave, ki jih moramo odpraviti. Potrebno je odkriti, katere vrednosti parametrov posameznih metod vračajo optimalne rezultate (torej največja vrednost mere F pri danem številu glasov)– pri tem mislimo na število izbranih najbližjih sosedov. Prav tako smo naleteli na težavo pri samem času izvajanja iskanja primernih vsebin, saj se je izkazalo, da za povprečnega uporabnika, ki je v sistem že vnesel približno 300 ocen, sistem potrebuje za obdelavo preko 6 minut. Sistem je sposoben šele po šestih minutah vrniti rezultate ter uporabniku ponuditi izbrane vsebine. To ni sprejemljivo, zato v naslednjih poglavjih opisujemo možne rešitve.

4.6.1 Modularni sistem

Sistem, ki smo ga opisali v četrtem poglavju, je vračal zelo dobre rezultate, vendar je za procesiranje posameznega uporabnika potreboval preveč časa – do 6 minut. Ugotovili smo, da je ena izmed slabosti to, da smo sistem zasnovali tako, da v trenutku, ko se uporabnik prijavi vanj, šele začne s procesiranjem. Uporabnik mora najprej čakati, da sistem zgradi njegov model, ter šele nato poišče primerne vsebine.

Zato smo se odločili sistem razdeliti na več posameznih storitev - modulov in za vsak modul ugotoviti, kdaj in kolikokrat ga je potrebno izvajati.

4.6.1.1 Faze uporabniškega modeliranja

V poglavju 2.1 smo opisali, kako lahko uporabniško modeliranje delimo na fazo učenja / gradnje uporabnikovega modela in na fazo iskanja uporabniku prilagojenih vsebin. Sistema torej najprej razdelimo na ta dva procesa ter ju uporabimo kot ločena modula.

Vprašanje je, če je smiselno uporabnikov model graditi oz. posodabljati po vsaki novi interakciji s sistemom. V začetni fazi je bilo to neizbežno, saj se je uporabnikov profil zgradil

vsakič, ko se je prijavil v sistem kot del celotnega procesa iskanja primernih vsebin. Po premisleku smo ugotovili:

- Če uporabnik od prejšnje interakcije s sistemom ni podal nobene nove ocene, se bo ob posodobitvi njegov model spremenil samo ob predpostavki, da so potencialni sosedje podali ocene za vsebine, ki jih je uporabnik do sedaj ocenil.
- Tudi če je uporabnik v sistem podal novo oceno, je lahko vpliv te ocene zelo majhen in ne pomeni drastične spremembe uporabnikovega profila. Če je uporabnik, na primer že ocenil 100 vsebin ter sedaj oceni še eno, pomeni, da ta ocena vpliva manj kot 1% na zgradbo celotnega profila. Torej, več kot je že obstoječih ocen (pod pogojem da pri gradnji profila upoštevamo vse), manjši je pomen vsake nove ocene.

Na osnovi teh ugotovitev smo zaključili, da ni potrebno posodabljati uporabniškega modela po vsaki vneseni oceni, ampak zadostuje, če to storimo samo po vsakih **M** vnesenih ocenah. Tako lahko močno zmanjšamo količino časa, ki je potreben, da se posameznemu uporabniku priporoči vsebine.

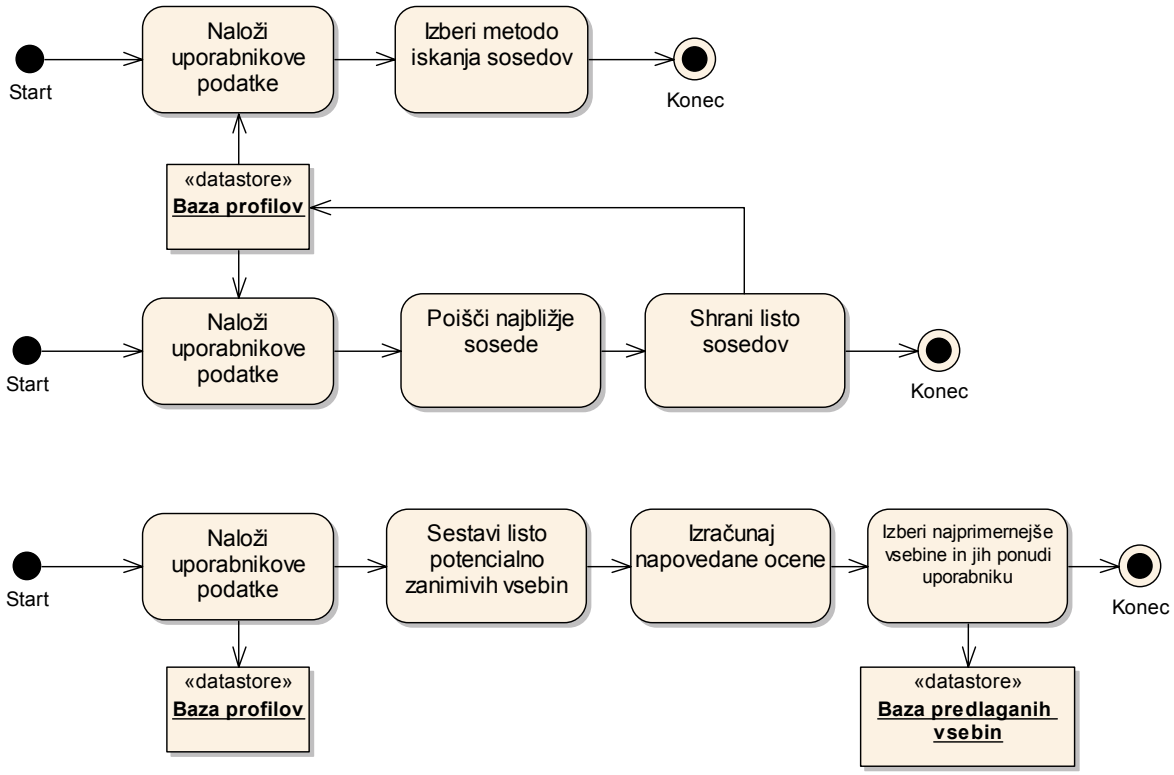
Ugotovili smo tudi, da je iskanje najbližjih sosedov aktivnega uporabnika časovno najbolj zahtevni korak celotne operacije. Z ločitvijo sistema na dva modula smo dosegli, da se uporabniški profil sedaj lahko posodobi, kadar uporabnik ni prijavljen v sistem. Ko se v novem sistemu uporabnik prijavi v sistem, se iz podatkovne baze naloži njegov trenutni profil in uporabi za iskanje primernih vsebin. Povprečni čas te operacije je manj kot sekunda, tako da uporabniku ni potrebno čakati.

4.6.1.2 Zasnova modularnega sistema

Pri delitvi sistema na posamezne module smo ohranili pristope iz sistema, opisanega v poglavju 4. Delitev na posamezne module smo dosegli na zelo enostaven način – s shranjevanjem vmesnih rezultatov v podatkovno bazo. Prva verzija sistema je namreč zgradila uporabniški model, z njegovo pomočjo poiskala primerne vsebine in jih nato ponudila uporabniku. Vsakič, ko se je uporabnik ponovno prijavil v sistem, se je celotna operacija ponovila od začetka. Za delitev sistema na module pa smo v sistem dodali funkcionalnost shranjevanja uporabniškega modela v podatkovno bazo.

Ob prijavi uporabnika v sistem se sedaj iz podatkovne baze naloži trenutni uporabniški model. Njegovi podatki se uporabijo kot nastavitve za iskanje primernih vsebin. Ob posodabljanju uporabnikovega profila pa sistem preveri, če profil že obstaja. V primeru, ko profil že obstaja, sistem samo posodobi vrednosti, če pa profil še ne obstaja, sistem zapiše nov vnos s primernimi podatki. Okolja, v katerem sistem deluje, nismo spreminjali in še vedno deluje v Java programskem okolju. Morali smo dodati funkcionalnosti vnosa in branja iz SQL podatkovne baze. V našem razvojnem okolju smo morali tudi postaviti strežnik s primerno podatkovno bazo.

Potek izvajanja posameznih modulov tega sistema je razviden iz slike 27.



Slika 27: Delitev skupinskega filtriranja na module

4.6.1.3 Uspešnost modularnega sistema

Cilj optimizacije sistema iz četrtega poglavja ni samo zmanjšanje samega števila operacij, potrebnih za iskanje uporabniku primernih vsebin, ampak tudi izboljšanje časovne zahtevnosti in uporabnikovega občutka pri delu s sistemom. Res je, da bi bilo možno z uporabo optimizacijskih postopkov s področja računalniških algoritmov in vpeljavo naprednejših programskih struktur število potrebnih operacij zmanjšati, vendar bi se brez delitve sistema na posamezne module vse operacije še vedno izvajale, kadar je uporabnik prijavljen v sistem. Zato smo ocenili, da je bolj pomembno, da se osredotočimo na časovno porazdelitev posameznih korakov.

Prva delitev je razmejitev na t.i. sprotno ter občasno fazo. V sprotni fazi se izvajajo samo kritične operacije, kot so iskanje primernih vsebin za aktivnega uporabnika, v občasni fazi pa posodabljanje uporabniškega profila. Možno bi bilo celo poiskati primerne vsebine v občasni fazi, vendar tega nismo izpeljali, ker se pri iskanju primernih vsebin ne bi upoštevale vse ocene najbližjih sosedov ter bi tako kako vsebino spregledali brez potrebe.

5. Izbira vrednosti parametrov sistema za skupinsko iskanje primernih vsebin

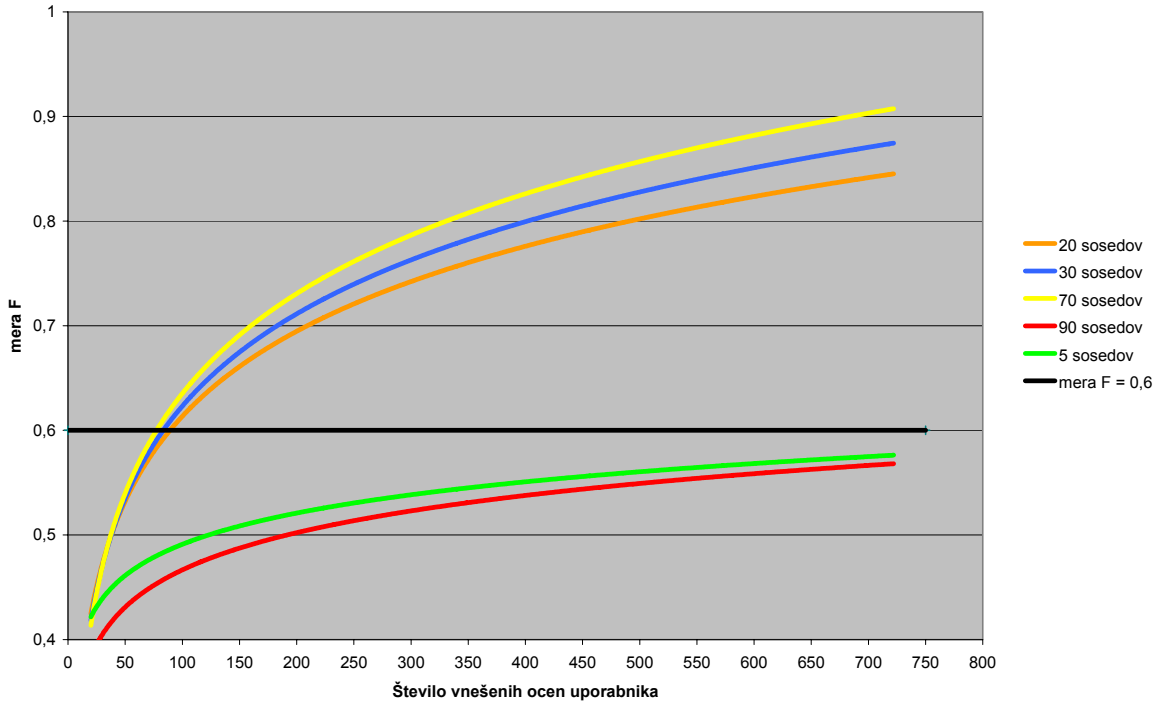
Sistem, opisan v poglavju 4, smo razvili in testirali ter ugotovili, da vrača sicer zelo dobre rezultate, vendar za posameznega uporabnika porabi preveč časa. Vsaka metoda, uporabljena v sistemu vsebuje lastne parametre, katerih vrednosti je potrebno pravilno nastaviti. Za vsako metodo je potrebno odkriti, koliko najbližjih sosedov potrebuje, da uporabniku najbolje išče primerne vsebine. Pri iskanju vsebin, zasnovanem na Bayesovem približku, pa je poleg števila sosedov potrebno tudi pravilno nastaviti vrednosti uteži, uporabljenih pri izračunu napovedane ocene.

5.1 Izbira pravilnega števila sosedov

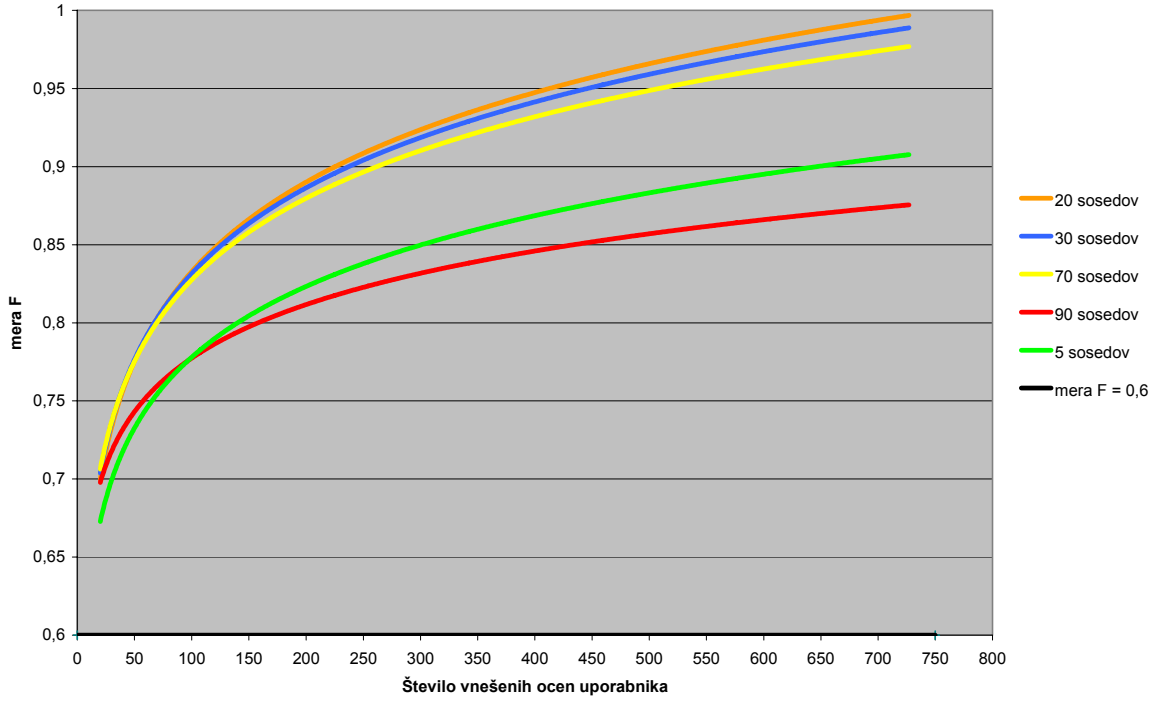
Pri sistemih osnovanih na principu skupinskega filtriranja vsebin je zelo pomembna izbira števila najbližjih sosedov **N**, ki jih mora sistem najti med gradnjo uporabniškega profila. Če sistem izbere preveliko število uporabnikov za najbližje sosede se lahko zgodi, da izbere tudi sosede, ki že preveč odstopajo od okusa dejanskega uporabnika. V takem primeru se lahko zgodi, da predlaga napačne vsebine ali pa izloči vsebine, ki bi bile v resnici pozitivno sprejete. V primeru majhnega števila najbližjih sosedov lahko trdimo, da je njihov okus dovolj podoben okusu aktivnega uporabnika, zato pa je izbira vsebin lahko premajhna. Manj sosedov lahko namreč pomeni tudi manjšo množico potencialnih vsebin, med katerimi sistem izbira. To pa lahko povzroči napačno izbiro oziroma izpuščene vsebine.

Za vsako od metod iskanja sosedov, opisanih v poglavju 4.3, smo želeli ugotoviti število sosedov **N**, pri katerem sistem najbolje deluje (najbolj zanesljivo išče primerne vsebine). Testiranje je potekalo v dveh korakih. V prvem koraku smo vse uporabniške modele posodobili tako, da so vsebovali izbrano število sosedov. V drugem koraku pa smo nato s pomočjo tako posodobljenih modelov poiskali primerne vsebine in izračunali napovedane ocene. Na podlagi napovedanih ocen smo nato izračunali vrednost mere F, s pomočjo katere vrednotimo uspešnost sistema.

5.1.1 Eachmovie



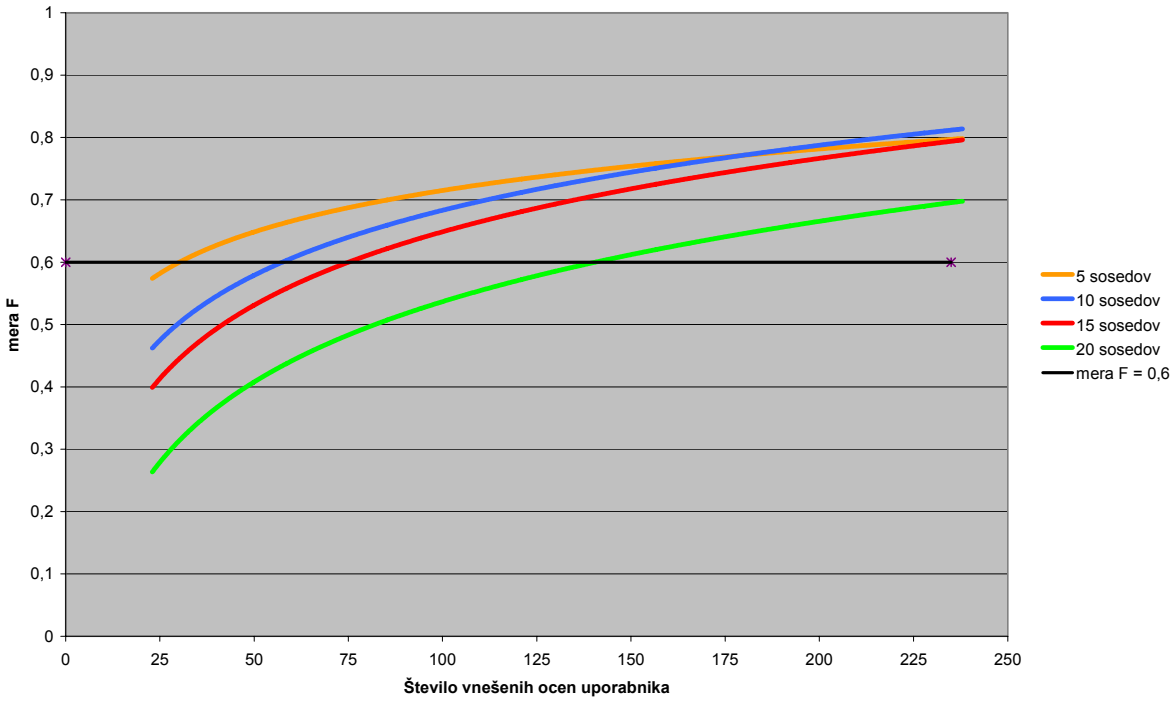
Slika 28: Vpliv števila sosedov – Pearsonov korelacijski koeficient - EachMovie



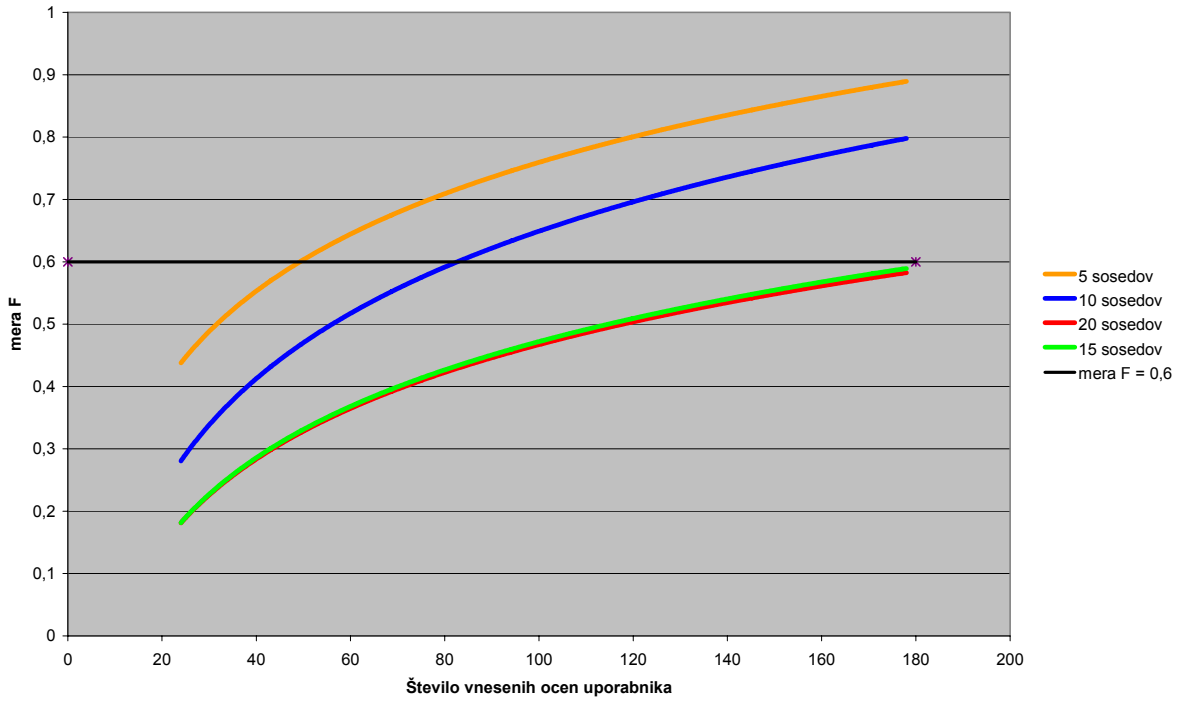
Slika 29: Vpliv števila sosedov – Evklidova razdalja - EachMovie

Iz rezultatov na slikah 28 in 29, je razvidno, da se rezultati metod glede na število izbranih sosedov zelo razlikujejo. Evklidova metoda za iskanje najbližjih sosedov doseže optimalne rezultate že pri 20 izbranih najbližjih sosedih. To je sorazmerno majhno število, glede na to, da je vseh potencialnih sosedov 64000. Pearsonov korelacijski koeficient optimalne rezultate vrača pri 70 izbranih najbližjih sosedih, kar je veliko več kot v primeru Evklidove distance. Za obe metodi je iz slik razvidno, da preveč (90) ali premalo (5) izbranih sosedov vodi do zelo slabih rezultatov, kot smo napovedali že v začetku poglavja. To je tudi razvidno iz rezultatov statistične primerjave potekov, prikazana v dodatku B.3. Ker bi bilo zelo neprimerno po vsakem preklopu uporabljene metode za iskanje sosedov ponovno graditi uporabniški model zaradi spremembe števila sosedov, je potrebno izdelati kompromisno rešitev. Za obe metodi smo zato izbrali 50 sosedov, saj vidimo, da Evklidovi metodi to ne predstavlja velike izgube kvalitete, pri Pearsonovem koeficientu pa dobimo skoraj najboljše rezultate.

5.1.2 Siol



Slika 30: Vpliv števila sosedov – Pearsonov korelacijski koeficient - SIOL



Slika 31: Vpliv števila sosedov – Evklidova razdalja - SIOL

Rezultati, prikazani na slikah 30 in 31, se močno razlikujejo od tistih, dobljenih pri Eachmovie podatkovni bazi. Kot vidimo iz rezultatov, obe metodi optimalno delujeta pri veliko manjšem številu izbranih sosedov. To je razvidno tudi iz rezultatov statistične primerjave prikazanih v dodatku B.4. Pri tej podatkovni množici lahko torej za obe metodi izberemo po pet najbližjih sosedov, ne da bi sklepali kompromise kot v primeru Eachmovie podatkovne množice.

5.1.3 Dokončna izbira optimalnega števila

Izkazalo se je, da optimalno število najbližjih sosedov za posamezno metodo ni fiksno nastavljeno, temveč je odvisno tudi od podatkovne množice oziroma okolja, v katerem se sistem uporablja. V vsakem sistemu je potrebno slediti lastnostim trenutne podatkovne množice in po potrebi prilagoditi obstoječe uporabniške modele. Sistem mora ob določenih intervalih ponovno sestaviti uporabniške modele vseh obstoječih uporabnikov. Pri odločanju o posodobitvi uporabniških modelov sta ključnega pomena naslednji lastnosti podatkovne baze:

- Število obstoječih uporabnikov – število izbranih najbližjih sosedov raste sorazmerno s številom obstoječih uporabnikov. Več kot je obstoječih uporabnikov, večje je lahko število izbranih najbližjih sosedov (če vmesni rezultati kažejo, da mera F pade pod izbrano mejo). Če se število obstoječih uporabnikov ni spremenilo, sprememba ni smiselna.
- Polnost podatkovne baze – več kot ima vsak uporabnik zabeleženih ocen, bolj natančno lahko sistem oceni njegovo primernost za najbližjega soseda. Bolj kot je polna podatkovna baza, natančnejše bo sistem izbral sosede. Vendar pa bo za iskanje predlogov potreboval več sosedov, da bo sploh lahko našel primerne vsebine. Če je

baza polna, pomeni, da je posamezen uporabnik ocenil večino obstoječih vsebin in je zato težje odkriti, katere primerne vsebine še obstajajo.

Sistem mora najprej na podlagi teh dveh lastnosti ugotoviti, ali je smiselno razmišljati o spremembi uporabniškega modela ali ne. V primeru, da sta se ti dve lastnosti od zadnje posodobitve dovolj spremenili (na primer, da je uporabnik v sistem dodal več kot 10 novih ocen ali pa da se je v celotnem sistemu zabeležilo več kot 1000 novih glasov), sistem sproži nov korak posodobitve in preizkusi, katero število sosedov je novi optimum. Ko to odkrije, posodobi vse obstoječe uporabniške modele in jim poišče primerno število najbližjih sosedov.

5.2 Izbira uteži za Bayesov približek

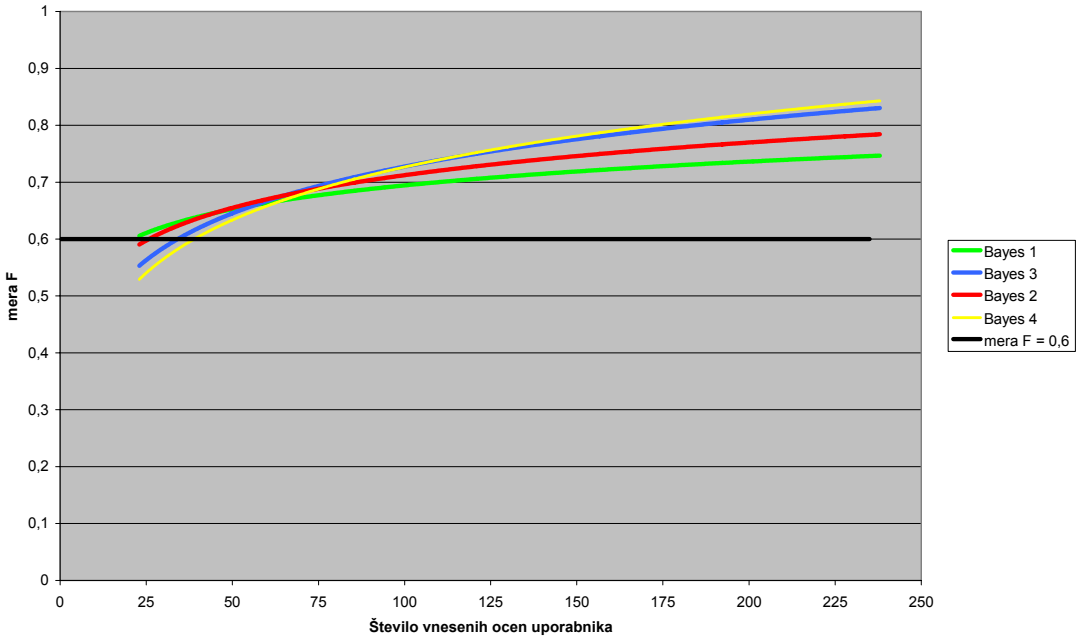
Pri delu z Bayesovim približkom, ki smo ga podrobneje opisali v poglavju 4.4.2, igra pomembno vlogo tudi izbira vrednosti uteži m (glej enačbo 4.12). Vrednost uteži določa, koliko na izračun primernosti vsebine vpliva povprečna ocena celotne publike $\bar{e}_p$ in koliko povprečna ocena s strani sosedov $\bar{e}_s$.

Če smatramo za pomembnejšo povprečno oceno širše publike $\bar{e}_p$, postane sistem manj prilagojen posameznemu uporabniku, saj bo predlagal večinoma vsebine, ki so všeč večini. Na ta način bi sicer sistem v večini primerov lahko predlagal vsebine, ki bi jih uporabnik pozitivno sprejel, razen če gre za uporabnika, ki ima zelo drugačen okus. Prednost velikega poudarka povprečne ocene širše publike je torej v tem, da lahko tak sistem bolj 'površno' išče najbližje sosede, saj ve, da bo povprečna ocena najbližjih sosedov manj vplivala na končni izračun primernosti posamezne vsebine.

V primeru, ko smatramo za pomembnejšo povprečno oceno najbližjih sosedov $\bar{e}_s$, pa sistem postane bolj prilagojen posameznemu uporabniku in naj bi zato v večini primerov uporabniku predlagal bolj primerne vsebine in upošteval tudi uporabnike z bolj eksotičnim okusom. Vendar pa hkrati postane veliko pomembnejša natančnost pri iskanju najbližjih sosedov, saj je zdaj iskanje primernih vsebin primarno odvisno od ocen s strani najbližjih sosedov.

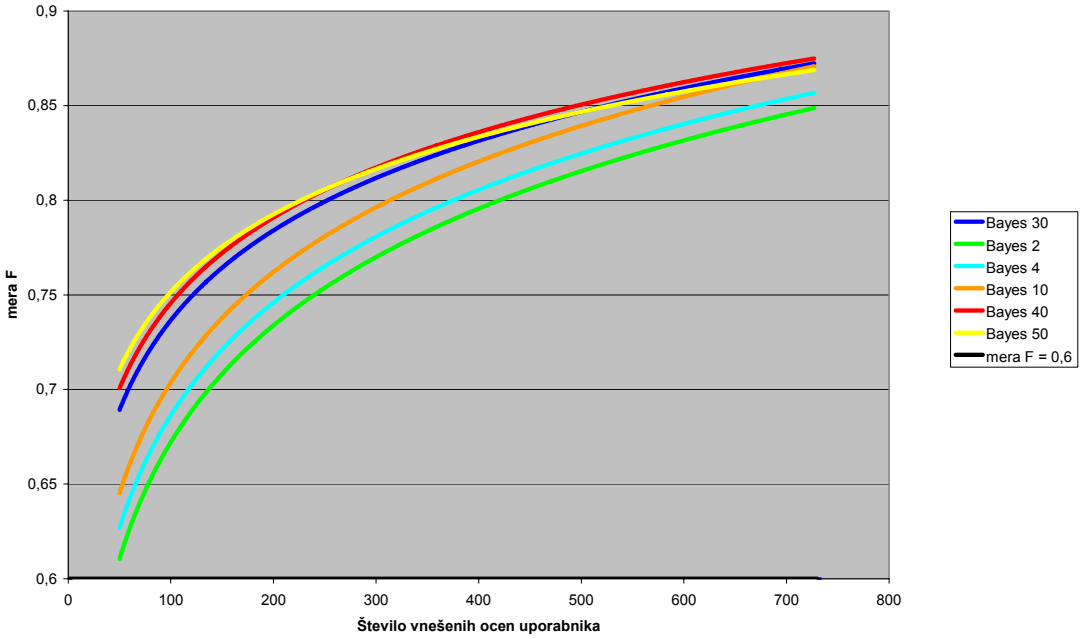
Rezultat poskusa lahko razložimo tudi drugače: ko ugotovimo, pri kateri uteži bo sistem najbolje deloval, bomo s tem tudi ugotovili, kolikšen delež najbližjih sosedov aktivnega uporabnika mora podati svoje mnenje o vsebini, da postane njihovo mnenje bolj verodostojno kot mnenje celotne publike. Če bo sistem optimalno deloval pri nizki uteži, to pomeni, da mora sistem bolj upoštevati mnenje najbližjih sosedov, tudi če se opredeli samo majhen delež le-teh. Velika utež pa pomeni ravno obratno – da bo sistem mnenje sosedov upošteval le tedaj, ko se bo jasno opredelila večina.

V okviru naših raziskav smo najprej zasnovali sistem za uporabniku prilagojeno iskanje vsebin, opisan v prejšnjem poglavju. Ko je sistem uspešno pričel delovati, smo se odločili za iskanje vsebin po metodi Bayesovega približka. Sistem smo preskusili za več različnih vrednosti uteži vrednotenja ocen širše publike in skušali analizirati, katere vrednosti dajejo optimalne rezultate. Za vsako izbrano vrednost uteži smo sistem zagnali in za vsako tako množico za vsakega uporabnika izračunali vrednosti mere F. Poskus smo izvedli na obeh podatkovnih množicah. Rezultati so razvidni na sliki 32:



Slika 32: Uteži bayesovega približka - SIOL

Iz slike 32, je razvidno, da sistem najbolje deluje takrat (najhitrejša rast mere F), ko je utež m nastavljena na vrednost $m = 4$ (rumeni potek), torej ko damo prednost oceni, podani s strani najbližjih sosedov šele takrat, ko je le-ta nastala na podlagi mnenja, ki ga je izrazila večina sosedov. V našem primeru vidimo, da je to v trenutku, ko so o vsebini svoje mnenje podali štirje od petih najbližjih sosedov. Vendar hkrati tudi vidimo, da to drži šele, ko je uporabnik v sistem vnesel več kot 75 ocen. V primeru manjšega števila vnesenih ocen pa sistem deluje podobno zanesljivo tudi, če upošteva oceno, ki je nastala na podlagi mnenja samo dveh ali celo enega soseda.



Slika 33: Uteži Bayesovega približka - EachMovie

Pri drugi podatkovni množici, prikazani na sliki 33, prav tako vidimo, da dobimo najboljši potek (najhitrejša rast mere F) v primeru, ko nastavimo Bayesovo utež na večjo vrednost, tako da zahtevamo, da več sosedov poda svoje mnenje o vsebini, preden upoštevamo napovedano oceno z njihove strani. To smo tudi preverili s pomočjo statistične primerjave, prikazane v dodatkih B.5 in B.6.

Rezultati poskusov, izpeljanih na obeh podatkovnih bazah, kažejo, da je v vsakem primeru koristno upoštevati ocene, ki so jih dali najbližji sosedi aktivnega uporabnika. Vendar pa lahko oceno sosedov upoštevamo kot primerno šele takrat, ko je nastala na podlagi izraženega mnenja večine sosedov – torej ko je oceno ocenilo več kot dve tretjini najbližjih sosedov.

6. Kombiniran vsebinsko-skupinski pristop

Pri delu s prvo verzijo sistema (ter kasneje z modularno verzijo) smo se osredotočili predvsem na preizkus obstoječih pristopov in na njihovo vgradnjo v sistem, ki bi vračal čim boljše rezultate. Uspeli smo izdelati sistem, ki vrača zadovoljive rezultate. Še vedno je nerešen problem števila potrebnih računskih operacij. Zato smo razvili nadgrajeno verzijo (porazdeljenega) sistema, kjer smo zmanjšali število operacij, potrebnih za posameznega uporabnika.

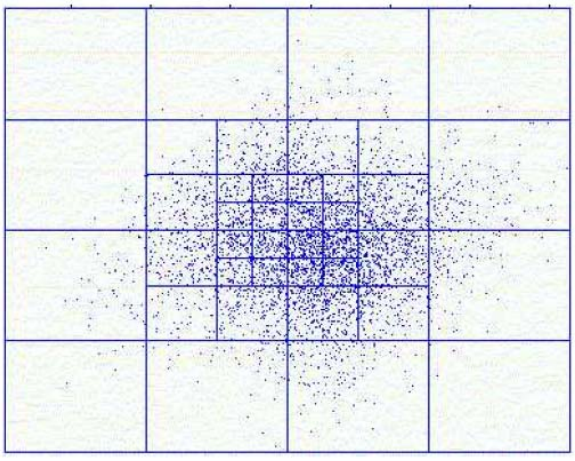
6.1 Osnovna ideja predlaganega pristopa k uporabniškemu modeliranju

Prvi verziji sistema sta bili osredotočeni na skupinske pristope. Glavna prednost teh pristopov je, da so popolnoma neodvisni od tipa vsebin, prisotnih v sistemu. Slaba stran takšne rešitve je, da je potrebno, če želimo uporabniku poiskati najbližje sosede, primerjati vsakega uporabnika z vsakim. Posledica je zelo veliko število potrebnih operacij, ki z vsakim dodanim uporabnikom kvadratično narašča.

Boljši način, ki bi zmanjšal število potrebnih operacij, bi bil predhodna delitev uporabnikov v gruče/skupine oziroma metode, ki bi nam omogočile, da najbližje sosede najdemo na drug, bolj enostaven način V prvi fazi poskusa smo zato želeli odkriti skupinski pristop, ki bi omogočil predhodno delitev uporabnikov v podskupine. Od obstoječih pristopov je najbolj izstopal pristop Eigentaste, zato smo se odločili, da ga najprej preizkusimo ter ugotovimo, ali je primeren za naše podatkovne množice.

6.2 Eigentaste

Algoritem Eigentaste [43] predstavlja pristop k razdelitvi uporabnikov v skupine ter iskanje primernih vsebin znotraj teh skupin. Pristop je osnovan na metodi glavnih komponent. Uporabnikom ob prvi prijavi v sistem ponudi najprej točno določen nabor vsebin in prosi za ocene. Na ta način ima sistem na voljo polno matriko ocen, ki vsebuje vse obstoječe uporabnike ter njihove ocene za začetne vsebine. Eigentaste nato izračuna lastne vrednosti in lastne vektorje te matrike z uporabo metode glavnih komponent. Zaradi ohranitve enostavnosti metoda izbere dva najbolj izrazita lastna vektorja ter nato uporabnike porazdeli po prostoru teh dveh lastnih vektorjev, kot je prikazano na sliki 34.

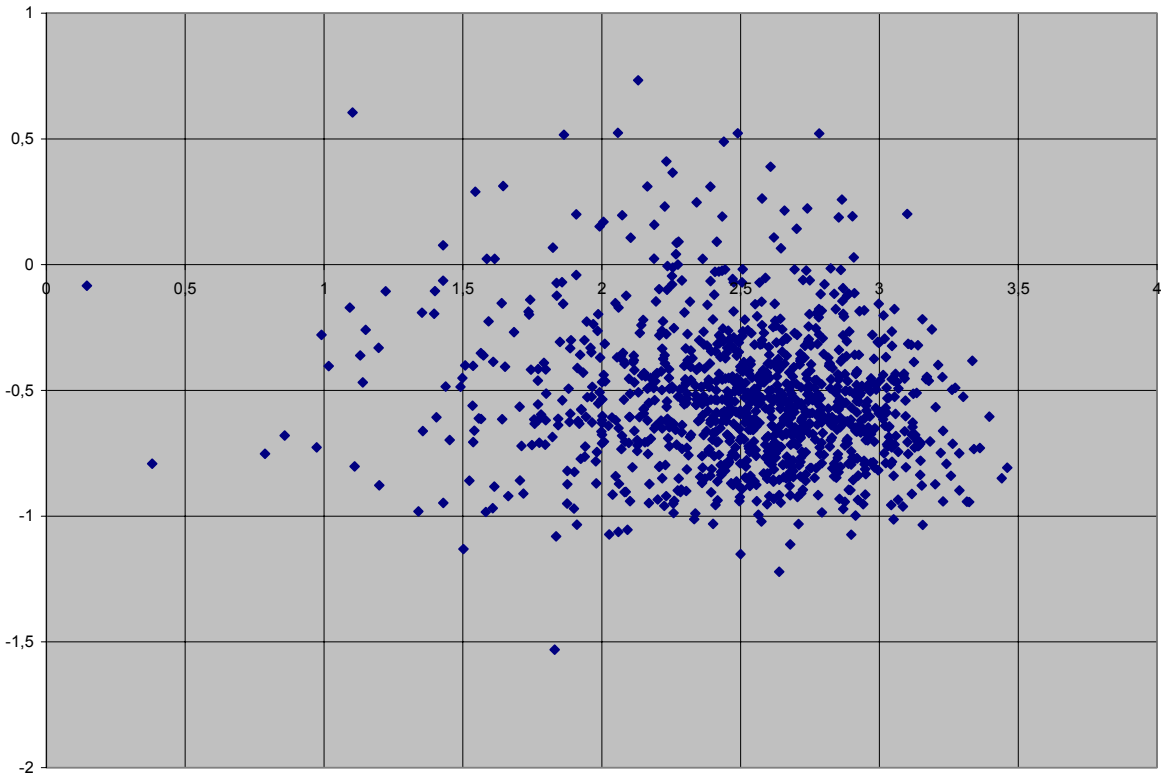


Slika 34: Porazdelitev uporabnikov z uporabo Eigentaste algoritma v originalnem članku

Vsaka točka na sliki predstavlja enega uporabnika oz. njegov okus. Zato lahko sklepamo, da so njegovi potencialni sosedje uporabniki, ki se v prostoru nahajajo v njegovi bližini. Eigentaste nato iskanje sosedov poenostavi tako, da prostor že predhodno razdeli na sektorje in podsektorje. Vsi uporabniki znotraj takšnega sektorja se nato obravnavajo kot najbližji sosedje drug drugega. Vsebine se nato predlaga glede na povprečno oceno, izračunano na podlagi ocen vseh uporabnikov znotraj skupine.

Eigentaste algoritem smo skušali vgraditi tudi v naše testno okolje. Prva težava, na katero smo naleteli še pred prvim zagonom algoritma, je ta, da sistem zahteva, da obstaja 20 vsebin, ki so jih ocenili vsi obstoječi uporabniki. Ker pri gradnji naših podatkovnih množic od uporabnikov ni bilo zahtevano, da ocenijo točno določene vsebine, je bilo za delo z Eigentaste algoritmom v našem sistemu potrebno računati lastne vrednosti matrike z manjkajočimi vrednostmi. Možnost, da bomo uporabnika dodelili v pravilno skupino, je bila manjša z vsako neocenjeno vsebino.

Da bi lahko kljub temu preizkusili primernost algoritma, smo iz podatkovne množice izločili podskupino uporabnikov, ki so vsi ocenili izbranih 15 vsebin. Na tej podskupini smo nato uporabili Eigentaste pristop. Prvi rezultati so prikazali, da je bila predstavitev porazdelitve uporabnikov po prostoru podobna Eigentastovi. Prostor smo nato razdelili na en nivo skupin (na 16 skupin), uporabnikom znotraj skupine poiskali sosede ter poiskali primerne vsebine. Na podlagi povratnih ocen smo nato izračunali uspešnost sistema. Razdelitev uporabnikov v skupine je prikazana na sliki 35.



Slika 35: Porazdelitev uporabnikov z uporabo Eigentaste algoritma na naši podatkovni množici

Rezultati dodeljevanja v skupine ter kasnejši rezultati predikcije so bili slabši kot tisti, objavljeni v članku o algoritmu. To je dokazovalo, da za našo podatkovno množico algoritem

ni primeren. Po podrobnejši analizi algoritma smo ugotovili, da je do težav prišlo predvsem zaradi izbire začetnih 'merilnih' vsebin (torej vsebin, ki jih uporabnik oceni ob prvi prijavi v sistem). Pri preizkusu Eigentaste algoritma v članku so bile začetne vsebine zelo natančno izbrane. Kriterij pri njihovi izbiri je bil, da so morale predstavljati različne pole možnih žanrov. Konkretno so bile pri preizkusu uporabljene šale iz različnih področij (politične, situacijske….). Začetna množica vsebin je bila določena tako, da je vsebovala natanko po dve šali iz vsake kategorije. Uporabnik je pri ocenjevanju začetne množice zelo natančno opredelil svoj okus za vsako kategorijo. V primeru naše podskupine nismo mogli vsebin izbirati, saj smo bili omejeni na iskanje vsebin, ki jih je ocenilo dovolj uporabnikov. Verjetno bi pri ponovitvi poskusa, kjer bi predhodno izbrali dovolj raznolike vsebine, dosegli boljše rezultate.

Dodatna težava takšnega pristopa je, da v okolju z multimedijskimi vsebinami (v našem primeru filmi) težko osnujemo pristop na ocenjevanju fiksno določenih vsebin, saj te vsebine zastarajo oziroma jih uporabniki sčasoma ne prepoznajo več.

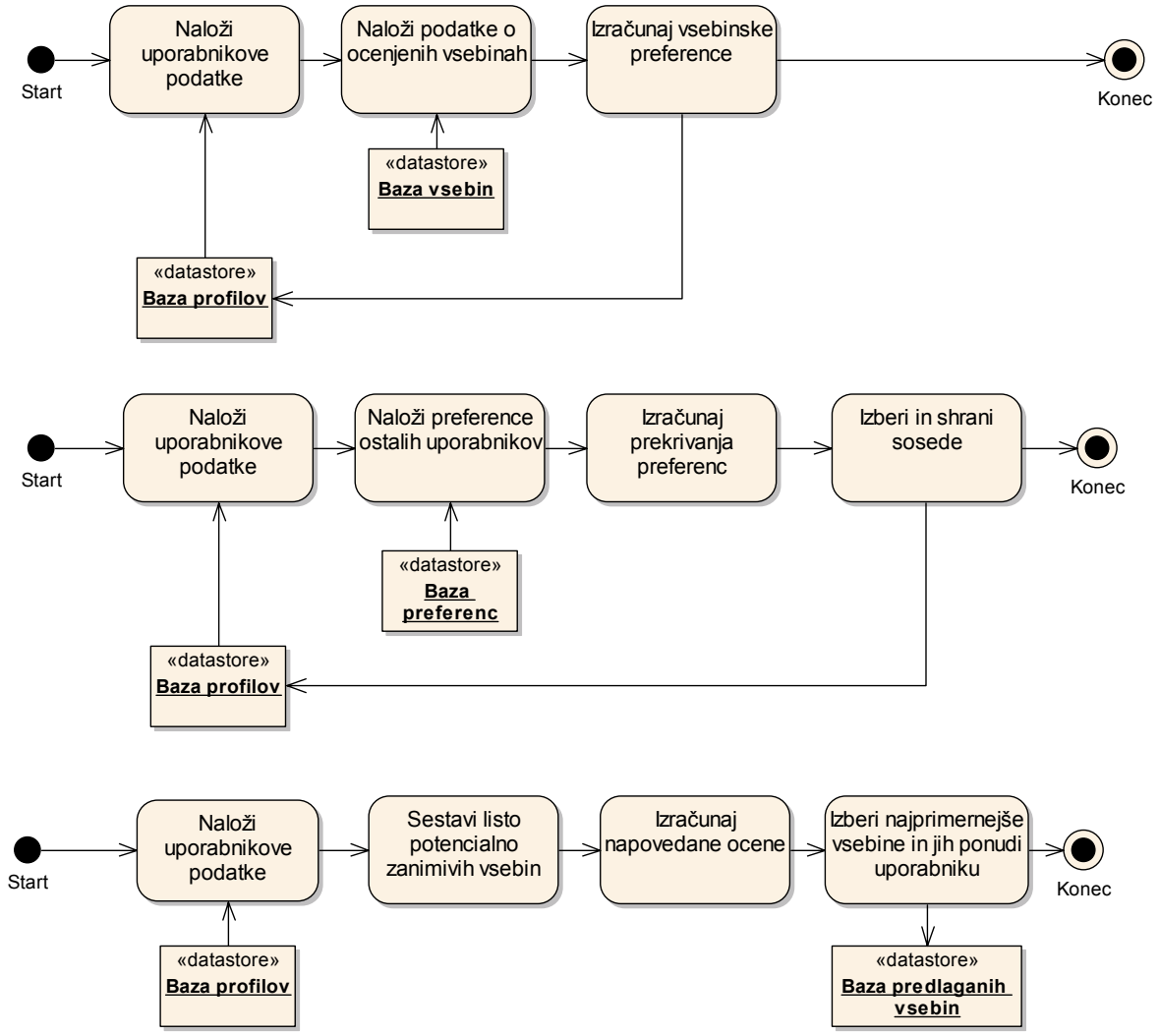
Test Eigentaste algoritma je pokazal, kako bi razvili pristop, ki je opisan v sledečem poglavju.

6.3 Razvoj novega pristopa za iskanje najbližjih sosedov

Na podlagi rezultatov pri preizkusu Eigentaste algoritma in podrobnejše preučitve same idejne zasnove algoritma smo se odločili razviti lasten, nov pristop. Ena od poglavitnih lastnosti Eigentasta je bila, da je algoritem deloval na osnovi skupinskega pristopa (torej z iskanjem sosedov aktivnega uporabnika), čeprav si je hkrati pomagal z vsebinskim pristopom. Vsebinski del je bil uporabljen samo v zasnovi sistema. Vsebinski del je bil uporabljen pri izbiri začetnih vsebin, ki so bile uporabniku ponujene pri prvi prijavi v sistem. Vsebine so bile izbrane tako, da so bile med seboj čimbolj raznolike, kar pa samo z uporabo skupinskega pristopa (ki zanemarja lastnosti vsebin) ni možno. Pri Eigentaste je bil vsebinski pristop uporabljen za predpripravo sistema. Ker Eigentaste kasneje v obratovanju uporablja samo skupinske pristope, se še vedno obravnava kot čisto skupinski pristop.

Izbrali smo podoben način uporabe vsebinskih pristopov v našem sistemu, ki bo bolj učinkovito iskal najbližje sosede [28, 45, 60]. Za razliko od Eigentasta smo se odločili vsebinske pristope uporabiti kot aktivni del sistema in ne samo ob sami postavitvi. Posledično se je naš sistem iz skupinskega spremenil v kombiniranega, ki vsebuje pristope iz obeh skupin in jih aktivno uporablja med delovanjem. Glavni razlog za tako odločitev je, da v sistemu, kjer se nove vsebine pojavljajo iz minute v minuto, ni možno določiti fiksnih vsebin, ki naj bi jih uporabnik ocenil ob prvem vstopu. Zato smo želeli s pomočjo vsebinskih pristopov sestaviti metodo, ki bi vedno omogočila iskanje najbližjih sosedov in bi jo bilo možno posodabljati ter prirejati glede na vse vsebine (ter njihove lastnosti) v podatkovni bazi.

Zamišljeni potek tega pristopa je prikazan na sliki 36:



Slika 36: Arhitektura kombiniranega vsebinsko-skupinskega sistema

Sistem najprej pregleda vse vsebine $h \in H$, za katere je aktivni uporabnik u_a v sistem že vnesel oceno (torej obstaja $e(u_a, h)$). Na podlagi analize metapodatkov teh vsebin $md(h)$ sistem nato nadgradi uporabnikov model z njegovimi vsebinskimi preferencami $ap(u_a, a_i)$, kar označimo s preslikavo $ap: A \times H \rightarrow AP$. $a_i \in A$ nam predstavlja vrednost atributa (glej 2.2). V našem primeru nam A predstavlja žanr, posamezni $a_i \in A$ pa eno izmed možnih vrednosti žanra (npr. $a_1 =$ akcija, $a_2 =$ komedija, $a_3 =$ fantastika….). Potencialne sosede sedaj iščemo na podlagi primerjajna vsebinskih preferenc posameznih uporabnikov. Na podlagi te primerjave lahko izračunamo podobnost dveh uporabnikov $sim(u_a, u_{ps})$. Kombinirani sistem, ki smo ga tako zasnovali, spada med kaskadne kombinirane sisteme (glej 2.4.3), ker se modula izvajata zaporedno; najprej s pomočjo vsebinskega modula poiščemo najbližje sosede, nato pa s pomočjo skupinskega pristopa uporabnikom poiščemo primerne vsebine.

Za skupinski del smo uporabili sistem, ki smo ga razvili že v prejšnji fazi raziskav in je bil podrobno opisan že v četrtem poglavju. Vsebinski del pa je bilo potrebno razviti na novo. Pri tem smo želeli doseči sledeče:

- Modul mora upoštevati obstoječe opise vsebin v podatkovni bazi.

- Obstajati mora možnost razširitve funkcionalnosti modula z dodajanjem novih tipov vsebin in s tem novih standardov opisov.

Ker je bil naš cilj predvsem zmanjšanje števila operacij, ki jih potrebuje skupinski modul, da najde primerne vsebine, smo se osredotočili na enostavnost sistema.

6.4 Gradnja vsebinskega profila

Naloga vsebinskega modula je, da z njegovo pomočjo ugotovimo preference uporabnikov glede na določene tipe vsebin in nato s pomočjo teh preferenc hitreje poiščemo primerne sosede. Ker se je pri preizkusu Eigentaste algoritma izkazalo, da delitev uporabnikov v fiksno določene skupine ne deluje učinkovito, smo se odločili za drugačen pristop. Prvi korak po vnosu novega uporabnika v sistem je postavitev njegovega profila na podlagi njegovih preferenc. Uporabniški model smo na novo razširili do te mere, da vsebuje dodatne informacije o njegovih vsebinskih preferencah.

Vodilna misel pri oblikovanju profila z vsebinskimi preferencami je enostavnost, ker nam bo ta del profila služil za iskanje sosedov, ne pa tudi za iskanje primernih vsebin. Zato ni potrebno, da vsebuje tako specifične podatke, kot bi jih potrebovali v popolno vsebinsko orientiranem sistemu. Velika težava, s katero se srečujejo vsebinski sistemi, je opis vsebine oz. podatki, ki so na voljo o posamezni vsebini. Opise vsebin, ki so nam bili na voljo, smo podali v poglavju 3.2. V našem sistemu smo uporabili samo polje, ki vsebuje podatke o žanru na osnovi sledečih zaključkov:

- Na osnovi literatnih virov sklepamo, da velika večina uporabnikov izoblikuje prvi vtis o vsebini na osnovi žanra [17].
- Žanr je eno redkih polj, katerega nabor vrednosti se redko ali celo nikoli ne spremeni.

Zelo pomembna je predvsem ugotovitev, da se žanr ne spremeni. Ker se nabor vrednosti žanra $a_i \in A$ le redko spreminja, lahko sistem zasnujemo na tej osnovi ter tako zelo pospešimo proces sestavljanja uporabniškega profila $up(u_a)$. V primeru, da bi uporabili polja, ki pogosto razširijo svoj nabor vrednosti (na primer igralec), bi nastala težava, ker bi morali sistem pred vsakim zagonom na novo učiti ter optimizirati parametre za izračun preferenc. Pomembno je, da to lahko naredimo predvsem zato, ker bo ta profil služil za iskanje sosedov in ne vsebin. Vsebine v naši podatkovni množici so lahko opisane z 23 žanri, ki so prikazani v sledeči tabeli:

Animacija	Avantura	Komedija	Družinski film	Akcija
Romantika	Drama	Kriminalka	Grozljivka	Biografija
Misterij	Triler	Fantazija	Znanstvena fantastika	Kratek film
Vojni film	Šport	Glasba	Dokumentarec	Zgodovinski
Musical	Vestern	Noir		

Tabela 4: Vrednosti atributa žanra a_i

Za vsako od možnih vrednosti izračunamo preferenco aktivnega uporabnika $ap(u_a, a_i)$. Lahko jo predstavimo kot številsko vrednost, ki zavzame tri možne vrednosti. Če ocenimo, da uporabnik izbrano vrednost žanra a sprejema pozitivno, nastavimo vrednost preference $ap(u_a, a_i)$ na 1, če reagira negativno na 0, v primeru, ko ni jasno opredeljen, pa na 0,5. Izračun preference $ap(u_a, a_i)$ je zelo enostaven – za vsako izmed možnih vrednosti atributa $a_i \in A$ izračunamo povprečno oceno $\bar{e}_a(u)$ na osnovi ocenjenih vsebin, opisanih s to vrednostjo. Ko imamo izračunano oceno, jo primerjamo z dvema mejnima vrednostima. Če je povprečna ocena vrednosti večja od zgornje mejne vrednosti, preferenco nastavimo na 1, ker sklepamo da uporabnik to vrednost žanra pozitivno sprejema. Če je povprečna ocena manjša od spodnje mejne vrednosti, nastavimo preferenco na 0, ker sklepamo, da uporabnik to vrednost žanra večino časa sprejema negativno. V vseh ostalih primerih pa vrednost preference nastavimo na 0.5, ker uporabnik ni dovolj jasno izrazil svoje preference. Psevodokda postopka za izračun preference je podan v dodatku A.5. Pri računanju povprečne vrednosti žanra upoštevamo samo ocene vsebin, ki v svojem opisu to vrednost tudi vsebujejo. Izračun ponovimo za vsako možno vrednost žanra in tako dobimo uporabniški profil, ki vsebuje preferenco za vsako izmed 23-ih vrednosti žanra.

6.5 Izbira primernih sosedov

V prejšnji verziji sistema (opisani v četrtem poglavju) je bilo pri iskanju primernih sosedov le-te primerjati z aktivnim uporabnikom na podlagi skupnih ocen ter s pomočjo izbranih pristopov izračunati utež, ki je predstavljala faktor ujemanja. Sedaj pa imamo na voljo uporabnikov vsebinski profil, ki vsebuje zgoščeno informacijo o vseh ocenah, ki jih je uporabnik do sedaj vnesel v sistem. Sedaj lahko namesto ocen primerjamo med seboj kar vsebinske profile. Sklicujemo se tudi na splošno trditev oziroma opis skupinskega pristopa, ki pravi, 'da skupinski pristopi uporabnikom poiščejo sosede s podobnim okusom'. Sosedje bi torej morali imeti tudi podoben vsebinski profil.

Namesto ocen bomo primerjali preference za posamezen tip žanra. Druga razlika je, da za primerjavo ni potreben kompleksen račun kot v prejšnjih primerih. Ker smo preference fiksno določili s tremi vrednostmi, lahko takoj pogledamo v koliko vrednostih se uporabnika ujemata in za sosede izberemo tiste z večjim številom ujemanj. Računanje primernosti se torej spremeni v štetje ujemanj.

Zaradi te poenostavitve je lahko problematična izbira postopka za iskanje primernih vsebin.

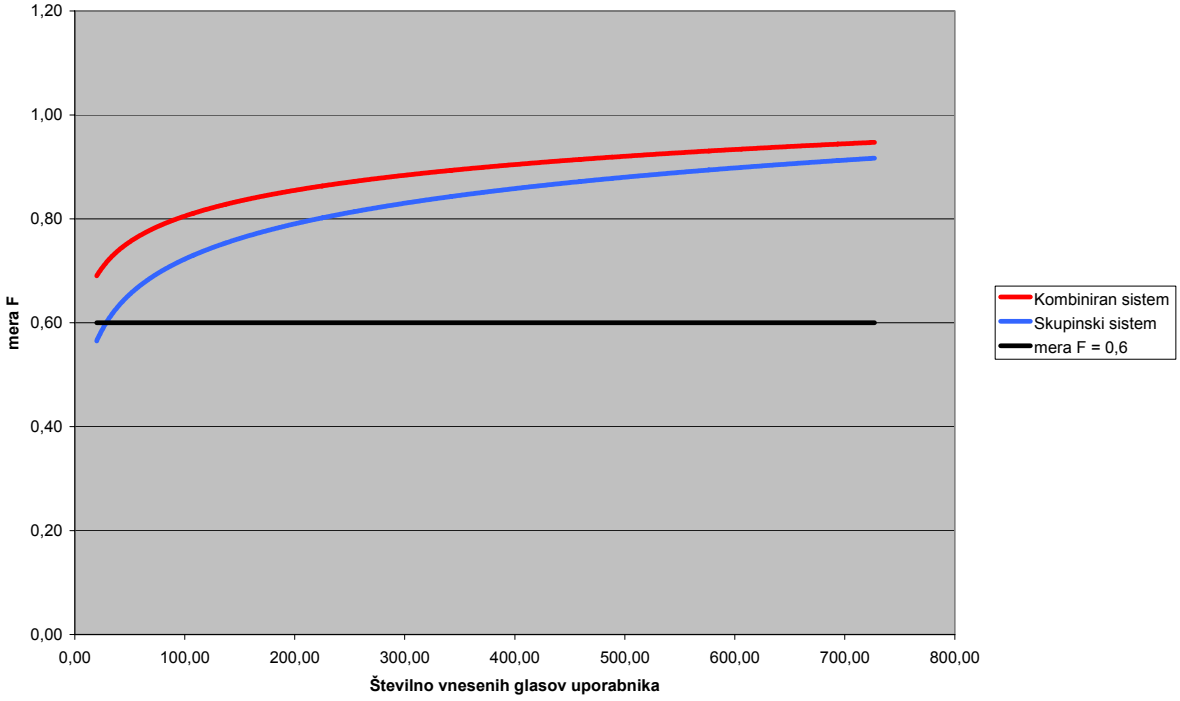
6.6 Iskanje primernih vsebin

Pri iskanju primernih vsebin je lahko težava, ker ne moremo uporabiti vseh pristopov, opisanih v prvotnem sistemu. Pristopa, ki sta osnovana na uteženi vsoti, namreč od nas zahtevata, da imamo za vsak par uporabnik-sosed izračunano mero podobnosti, ki se nato v samem algoritmu uporabi kot računska utež. Z uporabo našega novega načina iskanja sosedov pa imamo sedaj na voljo le število ujemanj, ki pa računsko ni ekvivalentno prej izračunani meri podobnosti. Možna rešitev je, da po izbiri primernih sosedov za vsakega izmed njih izračunamo utež s pomočjo postopkov iz prvotnega sistema. Rešitev ni ustrezna, ker v sistem vsiljuje dodatne module in operacije, ki jih načeloma ne potrebujemo. Zato smo problem raje rešili z uporabo Bayesovega približka, ki ne zahteva natančno izračunane podobnosti med uporabniki, poleg tega pa se je izkazal za najbolj uspešno metodo v sklopu prvih poskusov.

Postopek iskanja primernih vsebin je torej enak, kot je opisan v poglavju 4.4.2. To nam tudi omogoči objektivno primerjavo rezultatov obeh sistemov.

6.7 Rezultati kombiniranega sistema

Sistem smo uporabili na vseh obstoječih uporabnikih Eachmovie podatkovne množice. Vsakemu uporabniku smo izračunali njegove vsebinske preference, poiskali najbližje sosede in nato poiskali primerne vsebine. Vsaki vsebini smo izračunali napovedano oceno in jo nato primerjali z obstoječo oceno, zapisano v testni množici. Na podlagi teh primerjav smo izračunali natančnost, delež najdenih vsebin ter mero F. Rezultati so prikazani na sliki 37.



Slika 37: Primerjava obeh sistemov

Na sliki sta prikazani rezultatni krivulji kombiniranega in skupinskega sistema. Rdeča krivulja prikazuje potek rezultatov kombiniranega sistema, razvitega v šestem poglavju, modra pa skupinskega razvitega v četrtem poglavju. Iz grafa vidimo, da so rezultati kombiniranega sistema boljši, čeprav je mehanizem za iskanje sosedov teoretično bolj grob (ker sosedov ne primerjamo več na osnovi primerjavi vseh vnesenih ocen).

Statistična primerjava obeh sistemov (glej dodatek B.2) pokaže, da na začetku (manj kot 200 vnesenih glasov) kombiniran sistem deluje bolje. Ko je uporabnik vnesel več kot 200 glasov pa se oba sistema podobno obneseta.

Na osnovi teh primerjav lahko sklepamo, da tudi pri iskanju najbližjih sosedov lahko pride do prekomernega prileganja, če uporabimo preveč natančen pristop.

6.7.1 Računska zahtevnost

Podrobnejša analiza uporabljenih algoritmov je pokazala, da potrebuje kombinirani sistem veliko manj računskih operacij, da bi uporabniku našel primerne vsebine. V primerjavi s prvotno verzijo sistema (opisano v poglavju 4.2) vidimo, da uporabnike ne primerjamo več na podlagi vseh ocenjenih vsebin, temveč samo na podlagi njihovih uporabniških profilov.

Za izračun vsebinskih preferenc posameznega uporabnika potrebujemo $93n + \mathrm{O}(1)$ računskih operacij, kjer je m število vseh vsebin, za katere je uporabnik podal oceno. Pri iskanju primernih vsebin pa sistem opravi $\mathrm{O}(k)$ računskih operacij, kjer je k število vsebin, ki so jih sosedje ocenili, aktivni uporabnik pa ne. Iskanje najbližjih sosedov pa se sedaj izvede z eno samo operacijo, ker podatkovna baza, v kateri so hranjeni vsebinski profili, omogoča poizvedbo, ki vrne vse uporabnike z že izračunanim prekrivanjem. To je možno na osnovi prekrivanja, ki je izračunano na podlagi omejenega števila primerjav (vse uporabnike primerjamo v 23 izraženih preferencah) in ni več manjkajočih vrednosti, saj so vse preference izračunane in določene.

Podrobnejši potek algoritma je podan s psevdokodo v dodatku na strani 84.

7. Zaključek

V disertaciji smo predstavili testiranje več različnih metod s področja skupinskega filtriranja. Kot rezultat testiranja smo razvili novo metodo za iskanje primernih vsebin ter sistem za preklapljanje med različnimi načini iskanja najbližjih sosedov. Ker je bil končni sistem skupinskega filtriranja vsebin računsko zelo zahteven, smo razvili nov pristop – vpeljali smo profil žanrskih preferenc uporabnika ter ga uporabili za iskanje najbližjih sosedov.

Vse sisteme smo testirali na dveh velikih obstoječih podatkovnih množicah, ki sta nam omogočala testiranje v realnih razmerah, z velikim številom uporabnikov ter vsebin (glej 3.2). Podatkovni množici sta se med seboj močno razlikovali ter nam tako omogočili simuliranje več različnih situacij, predvsem simuliranje novega sistema, kjer uporabniki še niso podali veliko ocen, ter simuliranje starejšega sistema, kjer so uporabniki ocenili že veliko večino vsebin. Ta dva scenarija sta zelo pomembna – prvi pokaže, kako dobro se sistem obnese v začetnih razmerah, kjer nima na voljo veliko podatkov, drugi pa, kako se sistem obnese, ko je podatkovna množica že dokaj poln in je tako težko najti primerne vsebine, ki jih uporabnik še ni videl.

Kot mero vrednotenja za vse naše poskuse smo izbrali metodo mere F, ki je bila podrobneje predstavljena v poglavju 3.4.1. Kot mejno vrednost, torej vrednost, ki jo mora sistem preseči, da začne uspešno delovati, smo izbrali vrednost mere F 0.6. Vse rezultate testiranj smo nato vrednotili na osnovi podatka, kako hitro je bila vrednost dosežena ter kako hitro je vrednost naraščala po presegu te meje.

V prvi seriji poskusov smo se osredotočili na testiranje postopkov za iskanje najbližjih sosedov s sistemom za skupinsko filtriranje vsebin. Osredotočili smo se na postopka **Pearsonovega korelacijskega koeficienta** ter **Evklidovo razdaljo** (glej 4.3). Vsakega od teh postopkov smo programirali v naše testno okolje ter nato z njegovo pomočjo uporabnikom poiskali najbližje sosede. Za merjenje uspešnosti smo nato s pomočjo teh sosedov poiskali primerne vsebine, jim izračunali napovedane ocene ter na podlagi le-teh dobili potek vrednosti mere F. Prvi rezultati so pokazali, da pri primerjavi obeh metod ugotovimo, da se Evklidska razdalja bolje obnese, ker hitreje preseže prag mere F = 0.6 ter doseže višjo končno vrednost. Vendar pa so primerjave posameznih uporabnikov pokazale, da še vedno obstajajo primeri, kjer se je Pearsonov korelacijski koeficient bolje odrezal. Zato smo podrobneje preverili, kakšne so lastnosti uporabnikov, pri katerih to velja. Ugotovili smo, da to velja za uporabnike, ki zelo odstopajo od svoje povprečne ocene. Na podlagi te ugotovitve smo nato sestavili končno verzijo modula za iskanje najbližjih sosedov. Naš modul deluje tako, da vsebuje obe metodi iskanja najbližjih sosedov ter med njimi preklaplja na podlagi izračunanega odstopanja od povprečne ocene vsakega uporabnika posebej. Rezultati takega sistema so bili nedvomno boljši, saj je sistem že pri najmanjšem številu vnesenih ocen deloval z vrednostjo mere F, ki je bila večja od 0.6.

Ko je bila metoda iskanja najbližjih sosedov določena, smo se osredotočili na metodo iskanja uporabniku primernih vsebin. Podrobneje smo testirali metodo **utežene vsote** ter metodo **Bayesovega približka** (glej 4.4). Obe metodi smo testirali tako, da smo najprej poiskali najbližje sosede s prej razvitim modulom ter nato s pomočjo teh sosedov poiskali primerne vsebine. Za vsako od teh vsebin smo nato izračunali napovedane ocene ter na podlagi primerjave izračunanih ter dejanskih ocen izračunali vrednost mere F. Primerjava rezultatov posamezne metode kaže na to, da se metoda Bayesovega približka bolje obnese. Vendar metoda Bayesovega približka ni primerna za vse razmere, ker se poleg ocen, podanih s strani najbližjih sosedov, zanaša tudi na vse obstoječe ocene v sistemu. Če se torej sistem nahaja na začetku delovanja in nima veliko zabeleženih ocen, se lahko zgodi, da rezultati te metode niso

zanesljivi. Ker pa se je metoda utežene vsote zelo slabo obnesla, bi bila slab nadomestek za take primere. Zato smo se odločili, da razvijemo novo metodo, osnovano na metodi utežene vsote, ki bi delovala z večjo zanesljivostjo kot pa originalna metoda. Preučitev metode utežene vsote nam je pokazala, da daje ta metoda zelo velik pomen uporabnikovi povprečni oceni ter da vse vrednosti napovedanih ocen zasnuje na le-tej. Zato smo jo prilagodili tako, da te vrednosti sploh ne upošteva več, temveč se osredotoči na mnenje najbližjih sosedov. Rezultat te prilagoditve je bila metoda **prilagojene utežene vsote** (glej 4.4.3), ki je vračala veliko boljše rezultate kot osnovna metoda utežene vsote.

Ne glede na to, katero metodo izberemo, naletimo pri iskanju primernih vsebin na podlagi skupinskega filtriranja na pomembno vprašanje – koliko najbližjih sosedov moramo poiskati, da bo sistem najbolje deloval. Vsako izmed metod za iskanje primernih vsebin (utežena vsota, Bayeov približek in prilagojena utežena vsota) smo preizkusili z več različnimi števili izbranih sosedov. Vsakič smo najprej posodobili vse obstoječe uporabniške profile tako, da so vsebovali izbrano število najbližjih sosedov. Nato smo vsakemu uporabniku poiskali primerne vsebine (ločen poskus za vsako izmed treh metod) ter izračunali vrednosti mere F. Rezultati so bili zelo presenetljivi – ugotovili smo, da metoda, s katero iščemo nabljižje sosede, vpliva tudi na to, koliko sosedov potrebujemo, da sistem najbolje deluje (najhitreje doseže vrednost mere F = 0.6). Poleg tega smo ugotovili, da v vsakem primeru obstaja največje možno število najbližjih sosedov, pri katerem sistem še deluje. Če to število presežemo, začne sistem delovati vedno slabše (glej 5.1). Povrhu vsega pa na to, katero število sosedov je pravilno, močno vplivajo tudi lastnosti podatkovne množice, ki se uporablja v sistemu. Ker smo imeli možnost preizkuse izvajati na dveh različnih podatkovnih množicah, smo posumili, da 'polnost' podatkovne množice (torej kolikšen procent vseh možnih ocen je vnesen) močno vpliva na to število. Naš zaključek po izvajanju vseh preizkusov je, da število iskanih najbližjih sosedov ne sme biti fiksno določeno, temveč se mora s časom prilagajati spremembam v podatkovni množici. Če se procent zabeleženih ocen uporabnikov spremeni, je potrebno zagnati optimizacijski postopek, ki ugotovi, katero število najbližjih sosedov je v dani situaciji najbolj primerno, nakar se vsi uporabniški profili posodobijo.

Vzporedno z iskanjem pravilnega števila izbranih najbližjih sosedov smo iskali tudi pravilne vrednosti uteži pri metodi Bayesovega približka (glej 5.2). Metoda pri izračunu napovedane ocene daje prednost ocenam najbližjih sosedov ali ocenam vseh obstoječih uporabnikov. Kateri skupini daje prednost, je odvisno od vrednosti parametra – uteži. Metodo smo zagnali pri več različnih vrednostih uteži, vsakič izračunali vrednosti napovedanih ocen ter preko njih vrednosti mere F. Rezultati testiranja so nam pokazali podobno kot pri številu izbranih najbližjih sosedov – vrednost uteži ne sme biti fiksno nastavljena, temveč se mora spreminjati, kadar zaznamo, da se je procent zabeleženih ocen spremenil.

Na podlagi rezultatov testiranj smo torej ugotovili, da kombiniranje metod za iskanje najbližjih sosedov ter metoda Bayesovega približka zagotovijo najhitrejše naraščanje vrednosti mere F. Odkrili smo tudi slabost, s katero se večina strokovne literature ne ukvarja – računska zahtevnost. Medtem ko sistem vrača zelo dobre rezultate, se na žalost izkaže, da za iskanje primernih vsebin za vsakega uporabnika posebej porabi tudi do 6 minut. Ker to nikakor ni sprejemljivo, smo se odločili sistem izboljšati in tako težavo omiliti ali celo rešiti.

Kot prvo izboljšavo smo želeli uvesti skupine (gruče) s pomočjo postopka Eigentaste [43], ki naj bi se na tem področju dobro obnesel (glej 6.2). Naleteli smo na več težav, zaradi katerih smo postopek na koncu označili kot neprimeren za naše okolje. Prva težava je, da postopek za svoje delovanje zahteva obstoj 15-20 vsebin, ki so jih uporabniki ocenili ob registraciji. Te vsebine nam niso bile na voljo, kajti podatkovne množice, s katerimi smo sistem preizkušali, so bili vir sistemov, kjer uporabniki ob prvi prijavi ne ocenjujejo nobenih vsebin. Težavo smo delno odpravili tako, da smo iz podatkovne množice izbrali podmnožico uporabnikov, ki so se

v ocenah prekrivali pri 15 vsebinah. S pomočjo te podmnožice smo nato metodo Eigentaste uporabili na naših podatkovnih množicah. Rezultati so bili zelo slabi (redko so sploh presegli mejo mere F 0.6). Ko smo podrobneje preučili metodo, smo ugotovili, da ni primerna za naše okolje zaradi zahtev glede začetnih vsebin. Metoda je zasnovana tako, da od uporabnika pri prvi prijavi v sistem zahteva, da oceni 15 karakterističnih vsebin. Karakteristične vsebine so si med seboj čimbolj različne (primer na področju šal – ena šala o policajih, ena o blondinkah, ena politična itd.). V okviru filmskih vsebin, za katere smo snovali naš sistem, to ni možno doseči, ker se karakteristične vsebine s časom spreminjajo oz. izginjajo.

Kljub temu da metoda Eigentaste ni bila primerna za uporabo v našem okolju, pa je služila za razvoj novega pristopa. Eigentaste nam je pokazal, da bi bilo pametno naš sistem skupinskega filtriranja vsebin razširiti z dodajanjem nekaterih pristopov iz področja vsebinskega iskanja vsebin. Uvedli smo razširitev uporabniškega modela – uporabnikove preference za posamezen žanr (poglavje 6.3). Cilj teh preferenc je bil omogočiti primerjanje uporabnikov (in tako iskanje najbližjih sosedov) z manjšim številom operacij. Pred uvedbo preferenc smo dva uporabnika primerjali na podlagi vseh vsebin, ki sta jih oba ocenila, kar pomeni, da če sta ocenila večino obstoječih vsebin, je bilo potrebo izvesti temu primerno veliko število operacij. Z uvedbo preferenc pa uporabnike primerjamo samo na podlagi le-teh ter tako dva uporabnika vedno primerjamo samo na podlagi 23 preferenc (toliko različnih žanrov obstaja v naših podatkovnih množicah). Sistem, ki je najbližje sosede iskal na podlagi te metode, je deloval opazno hitreje, saj je poiskal primerne vsebine za vseh 64000 uporabnikov v manj kot enem dnevu. Poleg tega nas je presenetilo dejstvo, da je sistem deloval bolje kot pa naš originalni, čisto skupinski sistem za iskanje vsebin. Ugotovili smo, da je to posledica dejstva, da se sedaj najbližji sosedje z aktivnim uporabnikom prekrivajo v manj vsebinah ter ima sistem tako pri iskanju primernih vsebin na voljo večji izbor. Pri pravem skupinskem sistemu se namreč najbližji sosedje z aktivnim uporabnikom prekrivajo v sorazmerno velikem deležu vsebin (od 20% naprej) in tako večji del vsebin, ki so jih ocenili, odpade ko sistem preide v fazo iskanja primernih vsebin. V našem novem sistemu pa lahko prekrivanja sploh ni, saj smo se osredotočili na žanrske preference.

Sistem, ki smo ga tako razvili, deluje zelo hitro in zanesljivo. Poudariti je potrebno, da predstavlja tudi nov pristop h kombiniranim sistemom, saj ga ne moremo dodeliti v nobeno od skupin kombiniranih sistemov predstavljenih v poglavju 2.4.3. Koncept našega sistema je skupinsko filtriranje razširjeno s tehnikami iz vsebinskega filtriranja. Zaradi dodanih tehnik je sistem sedaj bolj omejen glede tipov vsebin – deluje lahko le v okolju z vsebinami, za katere imamo na voljo jasne opise, na podlagi katerih lahko izračunamo uporabnikove preference. Vendar imamo namen z nadaljnjim razvojem to slabost odpraviti in omogočiti uporabo sistema v vseh razmerah. Uvedba žanrskih preferenc nam je tudi omogočila interpretacijo rezultatov uporabniku, kar je zelo pomembno za kvaliteto uporabniške izkušnje pri uporabi sistema. Navaden skupinski sistem namreč ni sposoben uporabniku razložiti, zakaj predlaga kako vsebino oziroma lahko poda samo informacijo na podlagi, koliko ocen s strani najbližjih sosedov vsebino predlaga (npr. "Vsebino predlagam, ker jo je pozitivno ocenilo 5 od Vaših 15 sosedov"). Naš sistem pa lahko sedaj uporabniku poda razširjeno razlago na podlagi žanrskih preferenc, ki je uporabniku veliko bolj domača (npr. "Vsebino predlagam, ker je znanstvena fantastika in na podlagi Vaših dosedanjih ocen sem ugotovil, da vam ta žanr ustreza").

V nadaljnjem razvoju imamo namen naš sistem preizkusiti na podatkovnem setu Netflix [61]. Posebnost tega podatkovnega seta je, da je bil raziskovalcem ponujen kot izziv, kajti ponudniki so razvili svoj sistem za uporabniku prilagojeno iskanje vsebin. Zato ima podatkovni set še dodatno prednost ker nam bo omogočil primerjavo naših rezultatov z drugimi raziskovalnimi skupinami. Poleg tega bomo poskusili razširiti žanrske preference in tako poleg žanra podobnost med uporabniki računati tudi na podlagi drugih metapodatkovnih

polj. Na ta način bomo preskusili še primernost drugih polj in še dodatno razširili razlago, ki jo bo sistem podal uporabniku ob predlagani vsebini.

7.1 Izvirni prispevki k znanosti

V doktorski disertaciji smo predstavili nov način iskanja primernih vsebin ter uvedli nov in učinkovit način iskanja najbližjih sosedov. Posamezni koncepti v tem delu so izvirni prispevki k znanosti:

- **Prilagojena utežena vsota kot nova metoda za iskanje primernih vsebin za uporabnika**: Ugotovili smo, da izvorna metoda utežene vsote daje prevelik poudarek na povprečno oceno uporabnika, za katerega iščemo primerne vsebine. V primerih, kjer je ta povprečna ocena zelo velika ali pa zelo majhna, sistem nato ni sposoben najti pravilnih vsebin, ker ne upošteva mnenja najbližjih sosedov. V ta namen smo predstavili novo metodo – prilagojeno uteženo vsoto, ki povprečne ocene ne upošteva več. Tako razvita metoda se obnese bolje kot osnova metoda, kar smo prikazali v poglavju 4.4.3.
- **Sistemi za uporabniku prilagojeno iskanje vsebin ne smejo imeti fiksno nastavljenih parametrov**: Ugotovili smo, da so vrednosti parametrov pri katerih sistem najbolje deluje, odvisne od podatkovne množice, na kateri se uporablja. V ta namen smo poiskali optimalne vrednosti parametrov našega sistema za dve različni podatkovni množici. Ugotovili smo, da polnost podatkovne množice (torej število vnesenih ocen napram številu vseh možnih ocen) vpliva na to, pri katerem številu izbranih najbližjih sosedov sistem najbolje deluje. Bolj kot je podatkovna množica polna manj sosedov potrebujemo. Zato parametri sistema ne smejo biti fiksno nastavljeni, temveč se morajo prilagajati glede na trenutne lastnosti podatkovne množice. Testiranje vrednosti parametrov smo podrobneje predstavili v poglavju 5.
- **Kombiniran vsebinsko-skupinski sistem z žanrskimi preferencami**: Razvili smo nov kombiniran sistem, v katerem smo uvedli pomembno novost – žanrske preference. Sistem deluje po principu skupinskega filtriranja, torej da najprej poišče najbližje sosede ter nato na podlagi njihovih ocen poišče primerne vsebine. Vendar smo koncept iskanja najbližjih sosedov močno spremenili. Najbližjih sosedov sedaj ne iščemo več na podlagi posameznih ocen, temveč s pomočjo žanrskih preferenc, ki smo jih predhodno izračunali. Tako dobljen sistem deluje zelo hitro in bolje od navadnega sistema za skupinsko filtriranje vsebin. Zasnovo sistema priporočamo za vsa okolja, v katerih se večinoma ukvarjamo z enim samim tipom vsebin (npr. samo z filmskimi vsebinami) ter pričakujemo večje število uporabnikov. Razvoj in testiranje sistema smo podrobneje predstavili v poglavju 6.

Literatura

[1] G. Salton, M.J. McGill. An introduction to modern information retrieval. McGraw-Hill, 1983.

[2] D. Hand, H. Mannila, P. Smyth. Principles of data mining. Cambridge, The MIT Press, 2001.

[3] B. Crabtree, J. Soltysiak. Identifying and tracking changing interests. International journal of digital libraries, Vol. 2 , No. 1., str. 38-53, Springer Verlag, 1998.

[4] J. Mirković, D. Cvetković, N. Tomča in ostali. Genetic algorithms for intelligent internet Search, IEEE TCCA Newsletter, 1999.

[5] D. Mladenič. Text-learning and related intelligent agents: A survey. IEEE Intelligent systems, julij/avgust 1999.

[6] L. Ardissono, L. Console, I. Torre I. Exploiting user models for personalizing news presentations. proceedings of the 2nd workshop on adaptive systems and user modeling on the World Wide Web, 8th International World Wide Web Conference. Toronto, Canada, 1999.

[7] D. Billsus, M.J. Pazzani. A hybrid user model for news story classification. Proceedings of the Seventh International Conference on User Modeling, University of California, Pattaya, 1999.

[8] J. Konstan, B. Miller, D. Maltz in ostali. GroupLens: Applying collaborative filtering to Usenet news. Communications of the ACM, Vol. 40. str. 77-87, 1997.

[9] T. Malone, K. Grant, F. Turbak. Intelligent information sharing systems. Communications of the ACM, Vol. 30, str. 390-402, 1987.

[10] B. Bezzera, F. Carvahlo, G. Ramalho, J-D. Zucker. Speeding up recommender systems with-meta prototypes. Springer, Lecture Notes in computer science, advances in artifical intelligence, str. 227-236, 2002.

[11] Spletna stran trgovine Amazon.com (obiskana decembra 2008)

http://www.amazon.com

[12] A. Buczak , J. Zimmerman, K. Kurapati. Personalization: Improving ease-of-use, Trust and accuracy of a TV show recommender. In proceedings of the AH2002 Workshop on personalisation in future TV, Malaga, Španija, 2002.

[13] A. Difino, B. Negro, A. Chiarotto. A multi agent system for a personalized electronic program guide. In proceedings of the AH2002 Workshop on personalisation in future TV, Malaga, Španija, 2002, (elektronska verzija).

[14] K. Kurapati, S. Gutta, D. Schaffer, J. Martino, J. Zimmerman. A multi agent recommender. User Modeling 2001, Proceedings of 8th International Conference, UM 2001, Sonthofen, Springer, vol. 2109, 2001, (elektronska verzija).

[15] G. Anhanger, T. Little. Data semantics for improving retrieval performance of digital news video systems. IEEE Transactions on knowledge and data engineering, Vol. 13., No.3, 2001.

[16] G Uchygit, K Clark. An agent based electronic program guide. Proceedings of the AHA 2002 workshop on personalization, Malaga, Španija, 2002.

[17] J. Yuan, Y. Yu, X. Xiao, X. Li. SVM Based Classification Mapping for User Navigation. International Journal of Distributed Sensor Networks, Vol. 5, Issue 1, str. 23-32, 2009.

[18] M. Pogačnik. Uporabniku prilagojeno iskanje multimedijskih vsebin, doktorska disertacija. Univerza v Ljubljani, Fakulteta za elektrotehniko, 2004.

[19] NEC develops worlds first working prototype of mobile phone capable of receiving digital TV broadcasting. Spletna stran NEC – sporočilo za javnost, julij 2003. (elektronska verzija), http://www.nec.co.jp/press/en/0307/1001.html

[20] Spletna stran standarda Dublin Core http://dublincore.org (dostopano 5.2.2009).

[21] J.M. Martinez. MPEG-7 overwiev. International organisation for standardisation. 2004. spletna stran htttp://www.chiariglione.org/mpeg-7/mpeg-7.htm (dostopano 3.6.2005).

[22] Spletna stran TV-Anytime foruma http://tv-anytime.org (dostopano 5.2.2009).

[23] T. Požrl. Ontologije in njihova raba v multimedijskem okolju. Diplomska naloga. Univerza v Ljubljani, Fakulteta za elektrotehniko, 2005.

[24] Spletna stran TV-Anytime projekta, Metada Working Group http://www.tv-anytime.org/workinggroups/wg-md.html (dostopano 5.2.2009).

[25] Spletna stran projekta myTV http://www.hitech-projects.com/europrojects/mytv/ (dostopano 5.2.2009).

[26] Spletna stran standarda XML http://www.xml.com (dostopano 5.2.2009).

[27] B.S. Manjunath, P. Salembier, T. Sikore. Introduction to MPEG-7 Multimedia content description interface. John Wiley and Sons, Chichester, Anglija, 2002.

[28] R. Burke. Hybrid recommender systems: survey and experiments. User Modeling and User-Adapted Interaction, str. 331-379, 2002.

[29] M. Pogačnik, J. Tasič, M. Meža, A. Košir. Personal content recommender based on a hierarhical user model for the selection of tv programmes. User Modeling and User-Adapted Interaction, št. 15, str. 425-457, 2005.

[30] Spletna stran Youtube http://www.youtube.com (dostopano 2.2.2009).

[31] Spletna stran Shelfari http://www.shelfari.com (dostopano 12.2.2009).

[32] Spletna stran Facebook http://www.facebook.com (dostopano 12.2.2009).

[33] P. Cotter, B. Smyth. PTV: Intelligent personalized TV guides. Proceedings of the 12th Innovative Applications of Artificial Inteligence (IAAI-2000) Conference, Austin, Texas, str 957-964, 2000.

[34] M. Pogačnik, J. Tasič. Interactive and personalized television of the future. Proceedings of the First COST 267 workshop on Information and Knowledge Management for Integrated Media Communication, Firence, Italija, oktober 2001.

[35] E. Savia. Mathemathical Methods for a Personalized Information Service. Magistrsko delo, Helsinki University of Technology, Helsinki, Finska 1999.

[36] S. Raaijmakers, J. den Hartog, J. Baan. Multimodal topic segmentation and classification of news video. In proceedings of ICME 2002, vol. 2, starni 33-36, Lausanne, Švica, 2002.

[37] L. Ardissono, F. Portis, P. Torasso. Architecture of a system for the generation of personalized Electronic Program Guides. In proceedings of UM2001 Workshop on Personalization in Future TV, Sonthofen, Nemčija, 2001.

[38] C. Gena. Designing TV viewer stereotypes for an electronic program guide. In proceedings of the eighth international conference on user modelling, Sonthofen, Nemčija, 2001.

[39] K Kurapati, S Gutta. TV personalization through stereotypes. Proceedings of the AHA 2002 workshop on personalization, 2002.

[40] W. Cohen. Web collaborative filtering – Recommending music by crawling the web. Computer networks, Vol.33, No 1-6, str. 658-698, 2000.

[41] M. Setten, M. Veenstra, A. Nijholt. Prediction strategies: combining prediction techniques to optimise personalisation. In proceedings of the AH2002 workshop on personalization in future, Malaga, Španija, 2002.

[42] N. Pavešić. Razpoznavanje vzorcev. Univerza v Ljubljani, Fakulteta za elektrotehniko, 2000.

[43] K Goldberg, T Roeder, D Gupta, C Perkins. Eigentaste: a constant time collaborative filtering algoritm. UCB Electronics research technical report M00/41, University of California, Berkeley, avgust 2000.

[44] N. Good, B. Schafer, J. Konstan, A. Borchers, B. Sarwar, J. Herlocker, J. Riedl. Combining collaborative filtering with personal agents for better recommendations. American association for artificial intelligence, AAAI, str. 439-446, 1999.

[45] J. Salter, N. Antonopoulos. Cinemascreen recommender agent: combining collaborative and content-based filtering. IEEE Intelligent Systems, št. 21, str. 35-41, 2006.

[46] B. Mehta, W. Nejdl. Intelligent Distributed User Modelling: from semantics to Learning. Proceedings of User Modeling 2007, Krf, Grčija, 2007.

[47] Spletna stran programskega orodja XAMPP http://sourceforge.net/projects/xampp/ (dostopano 6.2.2009).

[48] Spletna stran programskega jezika Java http://java.com (dostopano 6.2.2009).

[49] Spletna stran programskega okolja Eclipse http://www.eclipse.org (dostopano 6.2.2009).

[50] EachMovie podatkovni set. Provided by Digital Equipment Corporation. http://research.compaq.com/SRC/eachmovie (dostopano 6.10.2005).

[51] MovieLens podatkovni set.

http://www.grouplens.org/node/73 (dostopano 20.11.2009).

[52] Spletna stran SiOL http://www.siol.net (dostopano 13.2.2009).

[53] J. Masthoff, The Pursuit of Satisfaction: Affective State in Group Recommender Systems. Proceedings on User Modeling 2005, Edinburgh, Velika Britanija, str. 297-306, 2005.

[54] M. Kunaver, T. Požrl, M. Pogačnik, J. Tasič, Optimisation of combined collaborative recomender systems. International Journal of Electronics and Communications (AEU), št. 61, str 433-443, 2007.

[55] E. S. Pearson. Student as a Statistician. Biometrika, vol 30, str. 210-250. Oxford University Press, Velika Britanija, 1939.

[56] R. H. C. Lopes, I. Reid, P.R. Hobson. The two dimensional Kolmogorov-Smirnov test. XI international workshop on Advanced Computing and Analysis Techniques in Physics Research Amsterdam, Nizozemska, 2007.

[57] F. Wilcoxon. Individual comparisons by ranking methods. Biometrics Bulletin, vol. 1, str 80-83, 1945.

[58] G. A. Miller. The Magical Number Seven, Plus or Minus Two: Some Limits in Our Capacity for processing Information. The Psychological Review, vol. 63, str. 81-97, 1956.

[59] Uradna spletna stran Internet Movie Database,

http://www.imdb.com (zadnji dostop junij 2003).

[60] M. Balabanović, Y. Shoham. Combining content based and collaborative recommendation. Communications of the ACM, Vol. 40, št. 3, 1997.

[61] Spletna stran podatkovnega seta Netflix, http://www.netflixprize.com (zadnji dostop november 2009).

[62] R. Johnsonbaugh, M. Schaefer. Algorithms. Pearson Education, Upper Saddle River, 2004.

[63] M. Zadravec, A. Brodnik, M. Mannila, M. Wanne, B. Žalik. A practical approach to the 2D incremental nearest-point problem suitable for different point distributions. Pattern Recognition Society, Elsevier, 41, str 646-653, 2008.

IZJAVA

Izjavljam, da sem doktorsko disertacijo izdelal samostojno pod vodstvom mentorja prof. dr. Jurija Tasiča ter somentorja prof. dr. Andreja Koširja. Izkazano pomoč drugih sodelavcev sem v celoti navedel v zahvali.

Matevž Kunaver

Dodatek A - psevdokoda

Pri razvoju našega sistema za uporabniku prilagojeno iskanje smo imeli opravka tudi s podatkovnimi strukturami in algoritmi. Tu bomo podrobneje opisali podatkovne strukture ter algoritme, s katerimi smo operirali.

A.1 UPORABNIŠKI MODEL

Uporabniški model nam predstavlja podatkovno strukturo, v kateri se hranijo vsi podatki, ki jih potrebuje sistem za prilagojeno iskanje vsebin.

Podatkovna struktura vsebuje sledeče podatke:

String userId

String sex

String age

String Zipcode

double[] ratings

array neighbours

double[] ratings je urejen seznam in tvori jedro uporabniškega modela. Predstavlja namreč zgodovino uporabnikovih dejanj – ocenjevanja vsebin. Posamezen element tega seznama vsebuje dva podatka – identifikacijo vsebine, ki jo je uporabnik oceni ter vrednost ocene.

array neighbours pa je urejen seznam, ki vsebuje podatke o uporabnikovih sosedih. To so ključni podatki, saj na podlagi njih sistem kasneje išče primerne vsebine. Posamezen element tega seznama je sestavljen iz dveh podatkov – identifikacije soseda ter izračunane podobnosti med sosedom in lastnikom profila.

Poleg standardnih funkcij za nastavljanje vrednosti spremenljivk v strukturi moramo opisati še:

getProfileData, ki iz uporabniškega profila prebere in vrne podatkovno strukturo raitngs.

setNeighbours, ki kot vhodno spremenljivko sprejme listo sosedov in podobnosti ter jo zapiše v urejeni seznam neighbours.

A.2 ALGORITEM ZA SKUPINSKO FILTRIRANJE VSEBIN

Prva verzija iskanja najprimernejših sosedov deluje tako, da najprej poišče listo vseh obstoječih uporabnikov. Nato najprej naloži podatke aktivnega uporabnika (torej uporabnika za katerega išče sosede). Na podlagi teh podatkov nato zaporedoma za vsakega uporabnika iz liste izračuna podobnost ter ga uvrsti na listo sosedov. To nadaljuje, dokler ni izračuna podobnosti za vse uporabnike na listi. Ko se ta korak zaključi, se lista sosedov uredi tako, da

so na vrhu uporabniki z največjo stopnjo podobnosti. Iz vrha liste se nato izbere samo **N** sosedov ter se jih zabeleži v profil aktivnega uporabnika kot njegove najbližje sosede.

Ker trenutna postavitev uporabljenih podatkovnih struktur ne omogoča iskanja vsebovanih ocen po identifikaciji vsebin, je bilo potrebno profile primerjati po načelu vsak-z-vsakim.

Input: Uporabniški profil aktivnega uporabnika, spisek vseh obstoječih uporabnikov, število (**N**) iskanih najbližjih sosedov.

Output: posodobljen uporabniški profil, ki ima sedaj vpisan tudi spisek najbližjih sosedov.

```
Input: ActiveUserProfile, PotentialNeighbourList, N
Output: ActiveUserProfile
NearestNeighbourSearch (ActiveUserProfile, PotentialNeighbourList, N, ActiveUserProfile) {
        NeighbourList = new list();
        MainRatings = ActiveUserProfile.getProfileData;
        int size = MainRatings.getSize();
        for (each neighbour on PotentialNeighbourList) {
                double[] tempSim = null;
                tempProfile = getNeighbourProfileFromDatabase(neighbourID);
                tempRating = tempProfile.getProfileData;
                for (each item found in MainRatings) {
                        match = checkIfListIncludesId(tempRating, MainRating.Id);
                        if (match != null)          {
                                tempSim.add(CalculateParitalSimilarity);
                        } MainRating.switchToNextItem();
                }
                if (tempSim != null) {
                        double Similarity = CalculateSimilarity(tempSim);
                        NeighbourList.add(NeighbourId, Similarity);
                }
                PotentialNeighbourList.switchToNextNeighbour();
        }
        NeighbourList.sort();
        list selection = NeighbourList.selectBestNeighbours(N);
        ActiveUserProfile.addNeigbours(selection);
        return ActiveUserProfile
}
```

Na podlagi zapisane psevdokode lahko tudi ocenimo računsko zahtevnost postopka [62, 63]. Predpostavimo, da sistem vsebuje podatke o **n** uporabnikih ter **m** vsebinah. Naš algoritem se izvaja v programskem okolju Java, zato je naša osnovna operacija en cikel Intelovega procesorja. V algoritmu se izvajajo tri *for* zanke, ena znotraj druge, zato ugotovimo, da za

procesiranje enega uporabnika sistem potrebuje $m^2 \cdot n$ operacij, ostale akcije pa v primerjavi s tem porabijo vsaj za en red manj računskih operacij (računska zahtevnost je torej $O(m^2 \cdot n)$).

A.3 ALGORITEM ZA ISKANJE PRIMERNIH VSEBIN

Algoritem za iskanje primernih vsebin mora najprej naložiti profil aktivnega uporabnika. V profilu uporabnika nato najde tabelo najbližjih sosedov ter v spomin naloži tudi profile le-teh. S pomočjo profilov najbližjih sosedov nato sestavi listo potencialno zanimivih vsebin (torej vseh vsebin, ki jih aktivni uporabnik še ni videl). Za vsako vsebino nato izračuna napovedano oceno ocene shrani v podatkovno bazo. Na koncu uporabniku ponudi izbor **M** vsebin.

Input: Uporabniški profil aktivnega uporabnika, število predlaganih vsebin

Output: Spisek predlaganih vsebin

```
Input: ActiveUserProfile, M
Output: RecommendedItemList
RecommendationGeneration (ActiveUserProfile, M) {
        NearestNeighbourList = ActiveUserProfile.getNeighbours();
        RecommendedItemList = new list();
        for (each neighbour on NearestNeighbourList){
                ratedItems = NeighbourProfile.getProfileData();
                for (each item in ratedItems){
                        match = checkIfItemIsAllreadyOntheList(RecommendedItemList, item);
                        if (match != null){
                                RecommendedItemList.addItem(item);
                        }ratedItems.switchToNextItem();
                }
        }
        MainRatings = ActiveUserProfile.getProfileData();
        for (each item in RecommendedItemList) {
                match = checkIfItemWasRatedByActiveUser(item, MainRatings);
                if (match != null) {
                        RecommendedItemList.removeItem(item);
                }
                else {
                        double predictedRating = calculatePredictedRating(item);
                        RecommendedItemList.addPredictedRating(item, predictedRating);
                }RecommendedItemList.switchToNextItem
        }
        RecommendedItemList.sortDescendingByPredictedRating;
        SelectionList = RecommendedItemList.selectTopRatedItems(M);
        return SelectionList;
```

```
}
```

A.4 ALGORITEM ZA PREKLOP MED MERAMI PODOBNOSTI

Algoritem za preklop med merami podobnosti služi izboljšanju rezultatov sistema za skupinsko filtriranje vsebin (glej 4.3.3). Za vhodne podatke sprejme uporabniški profil aktivnega uporabnika, nakar preveri lastnosti le-tega. Kot izhodni parameter vrne spremenljivko tipa boolean. To spremenljivko sistem kasneje interpretira pri izbiri metode za iskanje najbližjih sosedov (true pomeni, da sistem uporabi metodo Evklidove razdalje (glej 4.3.2), false pa metodo Pearsonovega korelacijskega koeficienta (glej 4.3.1)).

Input: Uporabniški profil aktivnega uporabnika, mejna vrednost za izbiro metode.

Output: spremenljivka tipa boolean

```
Input: ActiveUserProfile, treshold
Output: distanceSelector
MetricSwitcher (ActiveUserProfile, treshold) {
        boolean distanceSelector = true;
        MainRatings = ActiveUserProfile.getProfileData();
        double stdDev = MainRatings.calculateStandardDeviation();
        if (stdDev < treshold){
                distanceSelector = true;
        }
        else{
                distanceSelector = false;
        }
        return distanceSelector;
}
```

A.5 ALGORITEM ZA KOMBINIRAN VSEBINSKO-SKUPINSKI SISTEM

Jedro postopka, uporabljenega v kombiniranem sistemu, je algoritem za izračun uporabnikovih preferenc za posamezno vrednost žanra. Algoritem najprej naloži obstoječi profil uporabnika ter zgodovino vseh vsebin, ki jih je uporabnik do sedaj že ocenil. Sistem nato preveri, s katerimi žanri je opisana posamezna vsebina na spisku, ter oceno, s katero je bila ocenjena. Če je ocena višja od zgornjega praga, potem algoritem poveča preferenco za vsako vrednost žanra, s katero je vsebina opisana. V primeru, da je ocena nižja od spodnjega praga pa algoritem preference zmanjša. Ko algoritem tako preveri vse ocenjene vsebine, izdela tudi trenutni spisek preferenc za aktivnega uporabnika.

Input: Uporabniški profil Aktivnega uporabnika, uporabnikova zgodovina, profili ocenjenih vsebin, spisek možnih vrednosti žanra.

Output: posodobljen uporabniški profil, ki ima sedaj vpisan tudi spisek najbližjih sosedov.

```
Input: ActiveUserProfile, RatedItemList, GenreList
Output: GenrePreferencesProfile
GenrePreferencesCalculation (ActiveUserProfile, RatedItemList, GenreList) {
        GenrePreferencesProfile = new list();
        MainRatings = ActiveUserProfile.getProfileData;
        for (each item on RatedItemList) {
                for (each genreValue in GenreList) {
                        itemGenreProfile = item.getProfile();
                        if (itemGenreProfile.contains(genreValue) = true) {
                                if (rating > positiveTreshold) {
                                        GenrePreferencesProfile.increasePreference(genreValue);
                                }
                                else if (rating < negativeTreshold) {
                                        GenrePreferencesProfile.decreasePreference(genreValue);
                                }
                        }
                RatedItemList.switchToNexItem();
        }
        return GenrePreferencesProfile;
}
```

Na podlagi zapisane psevdokode lahko tudi ocenimo računsko zahtevnost postopka [53, 54]. Predpostavimo, da sistem vsebuje podatke o **n** vsebinah ter **m** žanrih. V algoritmu se izvajata dve *for* zanki, ena znotraj druge, zato ugotovimo, da za procesiranje enega uporabnika sistem potrebuje $m \cdot n$ operacij. Vendar pa je število možnih vrednosti žanra omejeno na 23, zato sistem za izdelavo profila posameznega uporabnika potrebuje samo $93n + O(1)$ računskih operacij.

Dodatek B – statistična analiza

Pri testiranju sistema smo za primerjanje uspešnosti različnih metod uporabili Mann-Whitney U statistični test [57]. V tem dodatku prikazujemo rezultate posameznih testiranj.

B.1 PRIMERJAVA USPEŠNOSTI METOD PREDIKCIJE (poglavje 4.5.1)

20 glasov

	Bayes	Utež. Vsota	p
povp.	0.5366	0.5825	0.755
std. dev.	0.2212	0.228	
N	920.0000	825	

	Bayes	Nova	p
povp.	0.5366	0.4855	0
std. dev.	0.2212	0.2335	
N	920.0000	861.0000	

	Utež. Vsota	Nova	p
povp.	0.5825	0.4855	0
std. dev.	0.228	0.2335	
N	825	861.0000	

50 glasov

	Bayes	Utež. Vsota	p
povp.	0.6511	0.6575	0.124
std. dev.	0.2188	0.2225	
N	435	400	

	Bayes	Nova	p
povp.	0.6511	0.5917	0
std. dev.	0.2188	0.2363	
N	435	432	

	Utež. Vsota	Nova	p
povp.	0.6575	0.5917	0.005
std. dev.	0.2225	0.2363	
N	400	432	

100 glasov

	Bayes	Utež. Vsota	p
povp.	0.6594	0.5964	0
std. dev.	0.2253	0.2053	
N	106	87	

	Bayes	Nova	p
povp.	0.6594	0.5726	0
std. dev.	0.2253	0.1932	
N	106	104	

	Utež. Vsota	Nova	p
povp.	0.5964	0.5726	0.192
std. dev.	0.2053	0.1932	

200 glasov

	Bayes	Utež. Vsota	p
povp.	0.7094	0.5732	0.002
std. dev.	0.2554	0.2417	
N	32	28	

	Bayes	Nova	p
povp.	0.7094	0.5255	0
std. dev.	0.2554	0.2524	
N	32	32	

	Utež. Vsota	Nova	p
povp.	0.5732	0.5255	0.731
std. dev.	0.2417	0.2524	

300 glasov

	Bayes	Utež. Vsota	p
povp.	0.5684	0.4793	0.099
std. dev.	0.2127	0.2541	
N	14	12	

	Bayes	Nova	p
povp.	0.5684	0.4748	0.118
std. dev.	0.2127	0.2475	
N	14	12	

	Utež. Vsota	Nova	p
povp.	0.4793	0.4748	0.9478
std. dev.	0.2541	0.2475	
N	12	12	

600 glasov

	Bayes	Utež. Vsota	p
povp.	0.75	0.4166	0.029
std. dev.	0	0.0962	
N	4	4	

	Bayes	Nova	p
povp.	0.75	0.3676	0.029
std. dev.	0	0.1528	
N	4	4	

	Utež. Vsota	Nova	p
povp.	0.4166	0.3676	0.657
std. dev.	0.0962	0.1528	
N	4	4	

B.2 PRIMERJAVA KOMBINIRANEGA SISTEMA IN DRUGEGA SISTEMA (poglavje 6.7)

20 glasov

	v1	v2	p
povp.	0.5366	0.709	0
std. dev.	0.212	0.1769	
N	920	453	

50 glasov

	v1	v2	p
povp.	0.6511	0.7826	0
std. dev.	0.2188	0.1722	
N	435	449	

100 glasov

	v1	v2	p
povp.	0.6594	0.8095	0
std. dev.	0.2253	0.1759	
N	106	59	

200 glasov

	v1	v2	p
povp.	0.7094	0.7674	0.222
std. dev.	0.2554	0.2337	
N	32	37	

300 glasov

	v1	v2	p
povp.	0.5684	0.788	0.729
std. dev.	0.2127	0.1996	
N	14	11	

600 glasov

	v1	v2	p
povp.	0.75	0.923	0.266
std. dev.	0	0.1087	
N	4	2	

B.3 IZBIRA SOSEDOV EACHMOVIE (poglavje 5.1.1

20 glasov

	5	**20**	p
povp.	0.5686	0.5403	0.031
std. dev.	0.2163	0.2193	
N	834	921	

	5	**30**	p
povp.	0.5686	0.5366	0.037
std. dev.	0.2163	0.2212	
N	834	920	

	5	**70**	p
povp.	0.5686	0.5359	0.042
std. dev.	0.2163	0.2231	
N	834	916	

	5	**90**	p
povp.	0.5686	0.594	0.038
std. dev.	0.2163	0.2317	
N	834	794	

	20	**30**	p
povp.	0.5403	0.5366	0.155
std. dev.	0.2193	0.2212	
N	921	920	

	20	**70**	p
povp.	0.5403	0.5359	0.09
std. dev.	0.2193	0.2231	
N	921	916	

	20	**90**	p
povp.	0.5403	0.594	0
std. dev.	0.2193	0.2317	
N	921	794	

	30	**70**	p
povp.	0.5366	0.5359	0.682
std. dev.	0.2212	0.2231	
N	920	916	

	30	**90**	p
povp.	0.5366	0.594	0
std. dev.	0.2212	0.2317	
N	920	794	

	70	**90**	p
povp.	0.5359	0.594	0.004
std. dev.	0.2231	0.2317	
N	916	794	

50 glasov

	5	**20**	p
povp.	0.6484	0.6366	0.609
std. dev.	0.2069	0.2259	
N	405	439	

	5	**30**	p
povp.	0.6484	0.6511	0.181
std. dev.	0.2069	0.2188	
N	405	435	

	5	**70**	p
povp.	0.6484	0.6442	0.27
std. dev.	0.2069	0.2306	
N	405	440	

	5	**90**	p
povp.	0.6484	0.6592	0.904
std. dev.	0.2069	0.223	
N	405	379	

	20	**30**	p
povp.	0.6366	0.6511	0.019
std. dev.	0.2259	0.2188	
N	439	435	

	20	**70**	p
povp.	0.6366	0.6442	0.092
std. dev.	0.2259	0.2306	
N	439	440	

	20	**90**	p
povp.	0.6366	0.6592	0.621
std. dev.	0.2259	0.223	
N	439	379	

	30	**70**	p
povp.	0.6511	0.6442	0.944
std. dev.	0.2188	0.2306	
N	435	440	

	30	**90**	p
povp.	0.6511	0.6592	0.174
std. dev.	0.2188	0.223	
N	435	379	

	70	**90**	p
povp.	0.6442	0.6592	0.182
std. dev.	0.2306	0.223	
N	440	379	

100 glasov

	5	20	p
povp.	0.5849	0.6871	0
std. dev.	0.2164	0.2181	
N	55	59	

	5	30	p
povp.	0.5849	0.6793	0
std. dev.	0.2164	0.2247	
N	55	60	

	5	70	p
povp.	0.5849	0.6955	0
std. dev.	0.2164	0.2011	
N	55	59	

	5	90	p
povp.	0.5849	0.5928	0.643
std. dev.	0.2164	0.2518	
N	55	47	

	20	30	p
povp.	0.6871	0.6793	0.979
std. dev.	0.2181	0.2247	
N	59	60	

	20	70	p
povp.	0.6871	0.6955	0.633
std. dev.	0.2181	0.2011	
N	59	59	

	20	90	p
povp.	0.6871	0.5928	0
std. dev.	0.2181	0.2518	
N	59	47	

	30	70	p
povp.	0.6793	0.6955	0.585
std. dev.	0.2247	0.2011	
N	60	59	

	30	90	p
povp.	0.6793	0.5928	0
std. dev.	0.2247	0.2518	
N	60	47	

	70	90	p
povp.	0.6955	0.5928	0
std. dev.	0.2011	0.2518	
N	59	47	

200 glasov

	5	20	p
povp.	0.5848	0.6981	0
std. dev.	0.2403	0.2492	
N	37	40	

	5	30	p
povp.	0.5848	0.7334	0
std. dev.	0.2403	0.2301	
N	37	41	

	5	70	p
povp.	0.5848	0.7152	0.001
std. dev.	0.2403	0.2447	
N	37	41	

	5	90	p
povp.	0.5848	0.6398	0.264
std. dev.	0.2403	0.2772	
N	37	35	

	20	30	p
povp.	0.6981	0.7334	0.063
std. dev.	0.2492	0.2301	
N	40	41	

	20	70	p
povp.	0.6981	0.7152	0.316
std. dev.	0.2492	0.2447	
N	40	41	

	20	90	p
povp.	0.6981	0.6398	0.026
std. dev.	0.2492	0.2772	
N	40	35	

	30	70	p
povp.	0.7334	0.7152	0.298
std. dev.	0.2301	0.2447	
N	41	41	

	30	90	p
povp.	0.7334	0.6398	0.003
std. dev.	0.2301	0.2772	
N	41	35	

	70	90	p
povp.	0.7152	0.6398	0.012
std. dev.	0.2447	0.2772	
N	41	35	

B.4 IZBIRA SOSEDOV SIOL (poglavje 5.1.2)

50 glasov

	5	**10**	p
povp.	0.4758	0.3642	0
std. dev.	0.1114	0.1241	
N	65	64	

	5	**15**	p
povp.	0.4758	0.3273	0
std. dev.	0.1114	0.1189	
N	65	64	

	5	**20**	p
povp.	0.4758	0.3551	0
std. dev.	0.1114	0.1473	
N	65	64	

	10	**15**	p
povp.	0.3642	0.3273	0.001
std. dev.	0.1241	0.1189	
N	64	64	

	10	**20**	p
povp.	0.3642	0.3551	0.847
std. dev.	0.1241	0.1473	
N	64	64	

	15	**20**	p
povp.	0.3273	0.3551	0.196
std. dev.	0.1189	0.1473	
N	64	64	

100 glasov

	5	**10**	p
povp.	0.6122	0.574	0.166
std. dev.	0.129	0.1559	
N	6	6	

	5	**15**	p
povp.	0.6122	0.562	0.367
std. dev.	0.129	0.1618	
N	6	6	

	5	**20**	p
povp.	0.6122	0.538	0.688
std. dev.	0.129	0.1342	
N	6	5	

	10	**15**	p
povp.	0.574	0.562	0.734
std. dev.	0.1559	0.1618	
N	6	6	

	10	**20**	p
povp.	0.574	0.538	0.922
std. dev.	0.1559	0.1342	
N	6	5	

	15	**20**	p
povp.	0.562	0.538	0.742
std. dev.	0.1618	0.1342	
N	6	5	

B.5 BAYES EACHMOVIE (poglavje 5.2)

50 glasov

	2	4	p
povp.	0.6492	0.6613	0
std. dev.	0.2086	0.207	
N	113	114	

	2	10	p
povp.	0.6492	0.6682	0.606
std. dev.	0.2086	0.2065	
N	113	115	

	2	30	p
povp.	0.6492	0.6985	0
std. dev.	0.2086	0.1909	
N	113	119	

	2	40	p
povp.	0.6492	0.6981	0.05
std. dev.	0.2086	0.1904	
N	113	121	

	2	50	p
povp.	0.6492	0.7052	0.03
std. dev.	0.2086	0.1833	
N	113	121	

	4	10	p
povp.	0.6613	0.6682	0.821
std. dev.	0.207	0.2065	
N	114	115	

	4	30	p
povp.	0.6613	0.6985	0
std. dev.	0.207	0.1909	
N	114	119	

	4	40	p
povp.	0.6613	0.6981	0.121
std. dev.	0.207	0.1904	
N	114	121	

	4	50	p
povp.	0.6613	0.7052	0.077
std. dev.	0.207	0.1833	
N	114	121	

	10	30	p
povp.	0.6682	0.6985	0.347
std. dev.	0.2065	0.1909	
N	115	119	

	10	40	p
povp.	0.6682	0.6981	0
std. dev.	0.2065	0.1904	
N	115	121	

	10	50	p
povp.	0.6682	0.7052	0
std. dev.	0.2065	0.1833	
N	115	121	

	30	40	p
povp.	0.6985	0.6981	0.862
std. dev.	0.1909	0.1904	
N	119	121	

	30	50	p
povp.	0.6985	0.7052	0.68
std. dev.	0.1909	0.1833	
N	119	121	

	40	50	p
povp.	0.6981	0.7052	0.085
std. dev.	0.1904	0.1833	
N	121	121	

100 glasov

	2	4	p
povp.	0.6799	0.7042	0.063
std. dev.	0.231	0.2193	
N	20	20	

	2	10	p
povp.	0.6799	0.7242	0.682
std. dev.	0.231	0.2036	
N	20	23	

	2	30	p
povp.	0.6799	0.7538	0.003
std. dev.	0.231	0.2257	
N	20	22	

	2	40	p
povp.	0.6799	0.7747	0.31
std. dev.	0.231	0.1869	
N	20	24	

	2	50	p
povp.	0.6799	0.7692	0.327
std. dev.	0.231	0.1867	
N	20	24	

	4	10	p
povp.	0.7042	0.7242	0.932
std. dev.	0.2193	0.2036	
N	20	23	

	4	30	p
povp.	0.7042	0.7538	0.003
std. dev.	0.2193	0.2257	
N	20	22	

	4	40	p
povp.	0.7042	0.7747	0.484
std. dev.	0.2193	0.1869	
N	20	24	

	4	50	p
povp.	0.7042	0.7692	0.518
std. dev.	0.2193	0.1867	
N	20	24	

	10	30	p
povp.	0.7242	0.7538	0.663
std. dev.	0.2036	0.2257	
N	23	22	

	10	40	p
povp.	0.7242	0.7747	0.001
std. dev.	0.2036	0.1869	
N	23	24	

	10	50	p
povp.	0.7242	0.7692	0.01
std. dev.	0.2036	0.1867	
N	23	24	

	30	40	p
povp.	0.7538	0.7747	0.892
std. dev.	0.2257	0.1869	
N	22	24	

	30	50	p
povp.	0.7538	0.7692	0.934
std. dev.	0.2257	0.1867	
N	22	24	

	40	50	p
povp.	0.7747	0.7692	0.559
std. dev.	0.1869	0.1867	
N	24	24	

B.6 BAYES SIOL (poglavje 5.2)

50 glasov

	1	2	p
povp.	0.6329	0.6345	0.748
std. dev.	0.1536	0.1549	
N	65	65	

	1	3	p
povp.	0.6329	0.6245	0.284
std. dev.	0.1536	0.1491	
N	65	65	

	1	4	p
povp.	0.6329	0.6105	0.04
std. dev.	0.1536	0.1353	
N	65	65	

	2	3	p
povp.	0.6345	0.6245	0.242
std. dev.	0.1549	0.1491	
N	65	65	

	2	4	p
povp.	0.6345	0.6105	0.692
std. dev.	0.1549	0.1353	
N	65	65	

	3	4	p
povp.	0.6245	0.6105	0.132
std. dev.	0.1491	0.1353	
N	65	65	

100 glasov

	1	2	p
povp.	0.6855	0.7452	0.047
std. dev.	0.1427	0.1191	
N	6	6	

	1	3	p
povp.	0.6855	0.7626	0.056
std. dev.	0.1427	0.1048	
N	6	6	

	1	4	p
povp.	0.6855	0.809	0.022
std. dev.	0.1427	0.0873	
N	6	6	

	2	3	p
povp.	0.7452	0.7626	0.337
std. dev.	0.1191	0.1048	
N	6	6	

	2	4	p
povp.	0.7452	0.809	0.03
std. dev.	0.1191	0.0873	
N	6	6	

	3	4	p
povp.	0.7626	0.809	0.07
std. dev.	0.1048	0.0873	
N	6	6	

Dodatek C – tabela matematičnih oznak in pojmov

OZNAKA	POMEN
u_i	Uporabnik *i*
h_k	Vsebina *k*
$e(u_i, h_k)$	Ocena, ki jo je uporabnik u_i dodelil vsebini h_k
$\hat{e}(u_i, h_k)$	Napovedana ocena, ki jo sistem izračuna za uporabnika u_i za vsebino h_k
$\overline{e}(u_i)$	Povprečna ocena uporabnika u_i
$\overline{e}(h_k)$	Povprečna ocena vsebine h_k
$a_i \in A$	Vrednost atributa A
$um(u_i)$	Uporabniški profil uporabnika u_i
$md(h_k)$	Metapodatkovni opis vsebine h_k
$ap(u_i, a_j)$	Preferenca uporabnika u_i za vrednost atributa a_j
$sim(u_i, u_j)$	Izračunana podobnost med uporabnikoma u_i in u_j
P	Natančnost (ang. *precision*)
R	Delež pravilno najdenih vsebin (ang. *recall*)
F	Mera F (ang. *F-measure*)